

LEVEL III

A106103

(12)

Report No. 4752

14

BBN-4752-VOL-3

AD A106104

6

**Application of Symbolic Processing to Command and Control:
An Advanced Information Presentation System.**

**Annual Technical Report No. 2
Volume III - Program Source Files.**

10

F/Zdybel, J/Gibbons, N/Greenfeld, and M/Yonke
Frank d Jess d Norton Martin

(12) 206

15 N00039-79-C-0316
ARPA Order-3740

11

August 1981

9 Annual technical
rept. no. 2, 1980-1981,

DTIC
ELECTE
OCT 27 1981
A

Prepared for:
Defense Advanced Research Projects Agency

This document has been approved
for public release and sale; its
distribution is unlimited.

DTIC FILE COPY

JOL

81 10 26 084

060100

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER	2. GOVT ACCESSION NO. AD-A106104	3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle) APPLICATION OF SYMBOLIC PROCESSING TO COMMAND & CONTROL: AN ADVANCED INFORMATION PRESENTATION SYSTEM. Vol.I: Overview, Vol.II: Knowledge Base, and Vol.III: Program Source Files		5. TYPE OF REPORT & PERIOD COVERED Annual Technical Report 1980 1981
7. AUTHOR(s) Frank Zdybel, Jeff Gibbons, Norton Greenfeld, and Martin Yonke		6. PERFORMING ORG. REPORT NUMBER BBN Report No. 4752
9. PERFORMING ORGANIZATION NAME AND ADDRESS Bolt Beranek and Newman Inc. 10 Moulton Street Cambridge, MA 02238		8. CONTRACT OR GRANT NUMBER(s) N00039-79-C-0316 ARPA Order No. 3740
11. CONTROLLING OFFICE NAME AND ADDRESS Defense Advanced Research Projects Agency Information Processing Techniques Office Arlington, VA 22209		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS
14. MONITORING AGENCY NAME & ADDRESS (if different from Controlling Office) Naval Electronics Systems Command Program Code 9D30 Washington, DC 20360		12. REPORT DATE August 1981
16. DISTRIBUTION STATEMENT (of this Report) Distribution of this document is unlimited.		13. NUMBER OF PAGES 30
17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report) Distribution of this document is unlimited.		15. SECURITY CLASS. (of this report) Unclassified
18. SUPPLEMENTARY NOTES		
19. KEY WORDS (Continue on reverse side if necessary and identify by block number) Information Presentation, Graphics, Knowledge Representation, Structured Inheritance Network, KL-ONE, Artificial Intelligence, Symbolic Processing, Computational Epistemology, Command and Control, AIPS		
20. ABSTRACT (Continue on reverse side if necessary and identify by block number) This report describes the work performed in the second year of the three-year contract to explore the application of symbolic processing to command and control (C2); specifically, the graphics interface between the C2 user and a complex C2 decision support system. <i>→ next page</i> In Volume I, the goals and approaches used in the design of the prototype system, AIPS (<u>A</u> dvanced <u>I</u> nformation <u>P</u> resentation <u>S</u> ystem), are discussed, as		

DD FORM 1473
1 JAN 73

EDITION OF 1 NOV 65 IS OBSOLETE

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

000100

Joc

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE(When Data Entered)

well as the year's efforts to extend the prototype. An overview of the current AIPS system is also provided.

Volume II contains the complete AIPS knowledge base. This document provides the fully-inherited structure that the system sees during operation.

Volume III contains the programs that manipulate the knowledge base and provide the active behavioral component of the system.



UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE(When Data Entered)

Report No. 4752

APPLICATION OF SYMBOLIC PROCESSING
TO COMMAND AND CONTROL:

AN ADVANCED INFORMATION PRESENTATION SYSTEM

Frank Zdybel, Jeff Gibbons, Norton Greenfeld and Martin Yonke

August 1981

Annual Technical Report #2
Volume 3 - Program Source Files

Prepared by:

Bolt Beranek and Newman Inc.
10 Moulton Street
Cambridge, Massachusetts 02238

Prepared for:

Defense Advanced Research Projects Agency
Information Processing Techniques Office
1400 Wilson Boulevard
Arlington, Virginia 22209

Accession For	
NTIS GRA&I	<input checked="checked" type="checkbox"/>
DTIC TAB	<input type="checkbox"/>
Unannounced	<input type="checkbox"/>
Justification	
By	
Distribution/	
Availability Codes	
Avail and/or	
Dist	Special
A	

This research has been supported by the Defense Advanced Research Projects Agency under Contract N00039-79-C-0316, ARPA Order No. 3740, and monitored by the Naval Electronics System Command (Program Code 9D30).

The views and conclusions contained in this paper are those of the authors and should not be interpreted as necessarily representing the official policies, either expressed or implied, of the Defense Advanced Research Projects Agency or the U.S. Government.

1. FUNCTION INDEX

The following contains an index to all functions in the AIPS source files. These files are generally organized as follows:

o Miscellaneous

AIPS	a file that merely loads all other files in the system
AIPSTEMP	a file of useful but otherwise uncategorizable functions
AIPSUTILITY	a file of utility routines

o Presentation

AIPSDISPLAY	the main presentation routines
AIPSCS	definition of coordinate systems
AIPSMAP	definition of the "map" presentation format
AIPSTABLE	definition of the "table" presentation format

o Realization

AIPSREALIZATION	the main realization modules
AIPSTODRAW	interface to the graphics primitives
AIPSDEVICES	a model of the graphics devices available
AIPSWINDOW	a simple window package in AIPS knowledge structures

o Domain Model

AIPSDOMAIN	the upper-level shared domain world structure
AIPSTS	general time system knowledge
AIPSGEOGRAPHY	general shared geographic knowledge
AIPSNAVAL	knowledge for depicting naval data
AIPSNTDS	specific knowledge for the NTDS presentation format

-- Index to Files --

<NEWAIPS>AIPS..3	8-Nov-80 10:49pm
<NEWAIPS>AIPSCS..10	3-Jul-80 5:03pm
<NEWAIPS>AIPSDEVICES..20	19-Jan-81 1:23am
<NEWAIPS>AIPSDISPLAY..47	19-Jan-81 10:43pm
<NEWAIPS>AIPSDOMAIN..5	10-Dec-80 11:41pm
<NEWAIPS>AIPSGEOGRAPHY..9	11-Dec-80 12:01am
<NEWAIPS>AIPSMAP..57	13-Nov-80 3:55am
<NEWAIPS>AIPSSNAVAL..3	17-Jul-80 2:14am
<NEWAIPS>AIPSNNTDS..26	13-Nov-80 2:21am
<NEWAIPS>AIPSRREALIZATION..28	29-Jan-81 1:10am
<NEWAIPS>AIPSTABLE..36	18-Dec-80 8:03am
<NEWAIPS>AIPSTEMP..111	2-Feb-81 3:43am
<NEWAIPS>AIPSTODRAW..63	2-Feb-81 3:12am
<NEWAIPS>AIPSTS..2	10-Jul-80 11:58pm
<NEWAIPS>AIPSUTILITY..9	19-Dec-80 12:34am
<NEWAIPS>AIPSWINDOW..122	2-Feb-81 1:15am

AIPSGetSpecifiedMetaDescriptions.....	AIPSDISPLAY	12
CollectTemplatesDescribingEntity.....	AIPSDISPLAY	13
DisplaySurfaceForGroundP.....	AIPSTODRAW	27
EstablishDomainEntityPositionInCS.....	AIPSMAP	23
FindApplicableApplicationSlots.....	AIPSDISPLAY	14
FindApplicationObjects.....	AIPSDISPLAY	15
FindColumnTableRows.....	AIPSTABLE	10
FindColumnWidth.....	AIPSTABLE	11
FindCostOfDisplay.....	AIPSDISPLAY	16
FindCoveringIntendedApplication.....	AIPSDISPLAY	17
FindDistanceSquaredFromDisplayForm.....	AIPSDISP	46
FindDomainCoordinateSystem.....	AIPSMAP	8
FindDrawingContextsForDisplay.....	AIPSTODRAW	28
FindMapEntitiesFromApplication.....	AIPSMAP	9
FindMapItemPrototypes.....	AIPSMAP	21
FindMappingBetweenCSs.....	AIPSMAP	22
FindMostSpecializedConcept.....	AIPSTEMP	17
FindRoleValuesTransitively.....	AIPSTODRAW	29
FindRowHeight.....	AIPSTABLE	12
FindRowTableColumns.....	AIPSTABLE	13
FindScaleUnitDistance.....	AIPSMAP	10
FindTableEntityIndicators.....	AIPSTABLE	14
FindTableThemeElementIndicators.....	AIPSTABLE	15
FindTableThemeElementIndicatorsAndOrientation...	IPSTABLE	16
FindTemplateObjects.....	AIPSDISPLAY	18
FindTemplateSlots.....	AIPSDISPLAY	19
FindTopLevelDisplaysOnViewSurface.....	AIPSTODRAW	30

GetFontCharacterDef.....	AIPSTODRAW	31
GuageTemplateRGCoverageOfTemplate.....	AIPSDISPLAY	20
HomogenousConceptListP.....	AIPSTEMP	18
Init2DCOORDINATESYSTEM.....	AIPSCS	1
Init2DLINERTRANSFIX.....	AIPSCS	2
Init2DLINERTRANSFORM.....	AIPSCS	3
InitAIPS.....	AIPSTEMP	1
InitAPERTURE.....	AIPSWINDOW	1
InitARRANGEMENT.....	AIPSTABLE	1
InitCARTESIANSYSTEM.....	AIPSCS	4
InitCIRCLE.....	AIPSREALIZATION	1
InitCLOSEDCURVE.....	AIPSREALIZATION	2
InitCOLUMN.....	AIPSTABLE	2
InitCOLUMNTABLE.....	AIPSTABLE	3
InitCommonDomainModel.....	AIPSDOMAIN	1
InitCONRACSCREEN.....	AIPSDEVICES	1
InitCOORDINATESYSTEM.....	AIPSCS	5
InitCoordinateSystemConcepts.....	AIPSCS	6
InitCTYWINDOW.....	AIPSDEVICES	2
InitCURVE.....	AIPSREALIZATION	3
InitDeviceConcepts.....	AIPSDEVICES	3
InitDevices.....	AIPSDEVICES	4
InitDISPLAY.....	AIPSDISPLAY	3
InitDISPLAYABSTRACTION.....	AIPSDISPLAY	4
InitDISPLAYATOM.....	AIPSREALIZATION	4
InitDISPLAYCOMPOSITE.....	AIPSREALIZATION	5
InitDISPLAYFORM.....	AIPSREALIZATION	6
InitDisplayFormConcepts.....	AIPSREALIZATION	7
InitDISPLAYITEM.....	AIPSTEMP	2
InitDISPLAYITEMABSTRACTION.....	AIPSTEMP	3
InitDISPLAYSURFACE.....	AIPSTEMP	4
InitDISTANCE.....	AIPSDOMAIN	2
InitDISTANCEUNIT.....	AIPSDOMAIN	3
InitEDGESET.....	AIPSREALIZATION	8
InitELLIPSE.....	AIPSREALIZATION	9
InitENEMYNTDSPLANE.....	AIPSNTDS	1
InitENEMYNTDSSHIP.....	AIPSNTDS	2
InitENEMYNTDSSUB.....	AIPSNTDS	3
InitENEMYPLANE.....	AIPSNAVAL	1
InitENEMYSHIP.....	AIPSNAVAL	2
InitENEMYSUB.....	AIPSNAVAL	3
InitFASTWINDOW.....	AIPSWINDOW	2
InitFRIENDLYNTDSPLANE.....	AIPSNTDS	4
InitFRIENDLYNTDSSHIP.....	AIPSNTDS	5
InitFRIENDLYNTDSSUB.....	AIPSNTDS	6
InitFRIENDLYPLANE.....	AIPSNAVAL	4
InitFRIENDLYSHIP.....	AIPSNAVAL	5
InitFRIENDLYSUB.....	AIPSNAVAL	6
InitGeographyModel.....	AIPSGEOGRAPHY	2

InitGEOREGION.....	AIPSGEOGRAPHY	1
InitHIDDENPLANE.....	AIPSDVICES	5
InitHOMOGENEOUSREGION.....	AIPSTEMP	5
InitINTEGERSYSTEM.....	AIPSCS	7
InitINTERVAL.....	AIPSDOMAIN	4
InitINTERVALUNIT.....	AIPSDOMAIN	5
InitINVISIBLESURFACE.....	AIPSTEMP	6
InitITEMPLATE.....	AIPSDISPLAY	5
InitLANDFEATURE.....	AIPSGEOGRAPHY	3
InitLANDMASS.....	AIPSGEOGRAPHY	4
InitLINEARTRANSFORM.....	AIPSCS	8
InitLINESEGMENT.....	AIPSREALIZATION	10
InitMAP.....	AIPSMAP	1
InitMapConcepts.....	AIPSMAP	6
InitMAPGRID.....	AIPSMAP	2
InitMAPITEM.....	AIPSMAP	3
InitMAPLEGEND.....	AIPSMAP	4
InitMAPPING.....	AIPSCS	9
InitMAPWITHBACKGROUND.....	AIPSMAP	5
InitMETRIC.....	AIPSDOMAIN	6
InitMiddleEarth.....	AIPSTEMP	8
InitMOTILEDISPLAYITEM.....	AIPSTEMP	7
InitNATION.....	AIPSGEOGRAPHY	5
InitNavalDomainModel.....	AIPSNAVAL	7
InitNONSCROLLWINDOW.....	AIPSWINDOW	3
InitNTDSConcepts.....	AIPSNTDS	7
InitNTDSENGAGEMENTMARK.....	AIPSNTDS	8
InitNTDSITEM.....	AIPSNTDS	9
InitNTDSMAP.....	AIPSNTDS	10
InitNTDSVELOCITYLEADER.....	AIPSNTDS	11
InitORTHOGONALSYSTEM.....	AIPSCS	10
InitPHYSOBJECT.....	AIPSDOMAIN	7
InitPLACE.....	AIPSDOMAIN	8
InitPLANE.....	AIPSNAVAL	8
InitPLANESURFACE.....	AIPSTEMP	9
InitPLATFORM.....	AIPSNAVAL	9
InitPOINT.....	AIPSREALIZATION	11
InitPOLYGON.....	AIPSREALIZATION	12
InitPOSITION.....	AIPSDOMAIN	9
InitPRESENTATION.....	AIPSDISPLAY	2
InitPresentationModel.....	AIPSDISPLAY	1
InitRealizationModel.....	AIPSREALIZATION	15
InitRECTANGLE.....	AIPSREALIZATION	13
InitRECTANGULARMAP.....	AIPSMAP	7
InitRECTANGULARREGION.....	AIPSTEMP	10
InitREGION.....	AIPSDOMAIN	10
InitREGULARPOLYGON.....	AIPSREALIZATION	14
InitROW.....	AIPSTABLE	4
InitROWTABLE.....	AIPSTABLE	5
InitSCROLLWINDOW.....	AIPSWINDOW	4
InitSHIP.....	AIPSNAVAL	10

InitSLOWWINDOW.....	AIPSWINDOW	5
InitSPEED.....	AIPSDOMAIN	11
InitSPEEDUNIT.....	AIPSDOMAIN	12
InitSTABILEDISPLAYITEM.....	AIPSTEMP	11
InitSUB.....	AIPSNAVAL	11
InitTABLE.....	AIPSTABLE	6
InitTableConcepts.....	AIPSTABLE	9
InitTEMPLATE.....	AIPSDISPLAY	6
InitTEXT.....	AIPSDISPLAY	7
InitTEXTCOLUMN.....	AIPSTABLE	7
InitTEXTROW.....	AIPSTABLE	8
InitTIME.....	AIPSDOMAIN	13
InitTIMESYSTEM.....	AIPSTS	1
InitTimeSystemConcepts.....	AIPSTS	2
InitTRANSFIX.....	AIPSCS	11
InitTRANSLATION.....	AIPSCS	12
InitTTYWINDOW.....	AIPSWINDOW	6
InitTURTLESCREEN.....	AIPSDEVICES	6
InitUNIT.....	AIPSDOMAIN	14
InitUNKNOWNNTDSPLANE.....	AIPSNTDS	12
InitUNKNOWNNTDSSHIP.....	AIPSNTDS	13
InitUNKNOWNNTDSSUB.....	AIPSNTDS	14
InitUNKNOWNPLANE.....	AIPSNAVAL	12
InitUNKNOWNSHIP.....	AIPSNAVAL	13
InitUNKNOWNSUB.....	AIPSNAVAL	14
InitUSERMODEL.....	AIPSTEMP	12
InitUserModel.....	AIPSTEMP	13
InitUT.....	AIPSTS	3
InitVEHICLE.....	AIPSDOMAIN	15
InitViewingOrganizationModel.....	AIPSTEMP	16
InitVIEWSURFACE.....	AIPSTEMP	14
InitVIEWSYSTEM.....	AIPSCS	13
InitVISIBLESURFACE.....	AIPSTEMP	15
InitWATERBODY.....	AIPSGEOGRAPHY	6
InitWATERFEATURE.....	AIPSGEOGRAPHY	7
InitWINDOW.....	AIPSWINDOW	7
InitWindowConcepts.....	AIPSWINDOW	8
InitWORLDPOSITION.....	AIPSGEOGRAPHY	8
LDIFFERENCE.....	AIPSTODRAW	32
MakeColumnString.....	AIPSDISPLAY	21
MakeDefaultTableGround.....	AIPSTABLE	17
MakeDefaultTextString.....	AIPSDISPLAY	8
MakeDisplayName.....	AIPSDISPLAY	9
MakeGroundForMapWithBackground.....	AIPSMAP	11
MakeLegendApplication.....	AIPSMAP	12
MakeLinearMapTransformAndGroundFromEntities.....	AIPSMAP	13
MakeLinearMapTransformFromGroundAndEntities.....	AIPSMAP	14
MakeMapItemApplication.....	AIPSMAP	24
MakeMapItemLabel?.....	AIPSMAP	15

MakeMapItems.....	AIPSMAP	16
MakeMapLabel.....	AIPSMAP	17
MakeMapTransformForMapWithBackground.....	AIPSMAP	18
MakeNTDSEngagementMark?.....	AIPSNTDS	15
MakeNTDSGroupStrengthMark?.....	AIPSNTDS	16
MakeNTDSVelocityLeader?.....	AIPSNTDS	17
MakeRectangle.....	AIPSREALIZATION	18
MakeRectangularMapBorder.....	AIPSMAP	19
MakeSimpleMapLegend.....	AIPSMAP	20
MakeTableColumnOrderList.....	AIPSTABLE	18
MakeTableEntryColumns.....	AIPSTABLE	19
MakeTableEntryRows.....	AIPSTABLE	20
MakeTableFoot?.....	AIPSTABLE	21
MakeTableHeaderRow.....	AIPSTABLE	22
MakeTableLeftIndex.....	AIPSTABLE	23
MakeTableRightIndex?.....	AIPSTABLE	24
MakeTableRowOrderList.....	AIPSTABLE	25
MakeTextHeight.....	AIPSDISPLAY	10
MakeTextWidth.....	AIPSDISPLAY	11
MapPositionDescription.....	AIPSMAP	25
OpenWindow.....	AIPSWINDOW	9
RotatePointsAroundPoint.....	AIPSTODRAW	34
ROUND.....	AIPSTODRAW	33
RoundPoint.....	AIPSTODRAW	35
RoundPoints.....	AIPSTODRAW	36
RptCoord.....	AIPSGEOGRAPHY	10
ScrollFastWindow.....	AIPSTODRAW	1
ScrollSlowWindow.....	AIPSTODRAW	2
ScrollTTYWindow.....	AIPSTODRAW	3
ScrollWindow.....	AIPSTODRAW	37
SelectDefaultDisplay.....	AIPSDISPLAY	22
SelectDisplay.....	AIPSDISPLAY	23
SelectFromAmongSufficientDisplays.....	AIPSDISPLAY	24
SendMessage.....	AIPSUTILITY	1
SetRoleValues.....	AIPSUTILITY	2
SupplyConceptMetaIndicator.....	AIPSDISPLAY	25
SupplyRoleMetaIndicator.....	AIPSDISPLAY	26
TDCircle.....	AIPSTODRAW	4
TDCurve.....	AIPSTODRAW	5
TDDisplay.....	AIPSTODRAW	6
TDDisplayComposite.....	AIPSTODRAW	7
TDDisplayItemAbstraction.....	AIPSTODRAW	8
TDDisplaySurface.....	AIPSTODRAW	9
TDEdgeSet.....	AIPSTODRAW	10
TDEllipse.....	AIPSTODRAW	11
TDFastWindow.....	AIPSTODRAW	12
TDIWFastWindow.....	AIPSTODRAW	13

TDIWNNonScrollWindow.....	AIPSTODRAW	14
TDIWSlowWindow.....	AIPSTODRAW	15
TDLineSegment.....	AIPSTODRAW	16
TDNonScrollWindow.....	AIPSTODRAW	17
TDPoint.....	AIPSTODRAW	18
TDRectangle.....	AIPSTODRAW	19
TDRegularPolygon.....	AIPSTODRAW	20
TDSlowWindow.....	AIPSTODRAW	21
TDText.....	AIPSTODRAW	23
TDTTYWindow.....	AIPSTODRAW	22
TDWindow.....	AIPSTODRAW	24
TEDisplayAtom.....	AIPSTODRAW	25
TEDisplayComposite.....	AIPSTODRAW	26
TemplateCGCoversTemplateCGP.....	AIPSDISPLAY	27
TemplateOutSpecifiesTemplateP.....	AIPSDISPLAY	28
TLAperture.....	AIPSWINDOW	10
TLColumnTable.....	AIPSTABLE	26
TLDisplay.....	AIPSDISPLAY	29
TLDisplayAtom.....	AIPSREALIZATION	16
TLDisplayItem.....	AIPSTEMP	19
TLDisplaySurface.....	AIPSTEMP	20
TLPlaneSurface.....	AIPSTEMP	21
TLRectangle.....	AIPSREALIZATION	17
TLRowTable.....	AIPSTABLE	27
TLWindow.....	AIPSWINDOW	11
TMAperture.....	AIPSWINDOW	12
TMDisplaySurface.....	AIPSTEMP	22
TMFastWindow.....	AIPSWINDOW	13
TMNonScrollWindow.....	AIPSWINDOW	14
TMRectangle.....	AIPSREALIZATION	19
TMRegionNumber.....	AIPSTEMP	23
TMSlowWindow.....	AIPSWINDOW	15
MTTTYWindow.....	AIPSWINDOW	16
TransformPoints.....	AIPSTODRAW	38
TransformSize.....	AIPSTODRAW	39
TSAperture.....	AIPSWINDOW	17
TSNonScrollWindow.....	AIPSWINDOW	18
TSPlaneSurface.....	AIPSTEMP	24
TSRectangle.....	AIPSREALIZATION	20
TSRectangularRegion.....	AIPSTEMP	25
TSWindow.....	AIPSWINDOW	19
TT2DLinearTransFix.....	AIPSCS	14
TT2DLinearTransform.....	AIPSCS	15
TTTransFix.....	AIPSCS	16
TTTranslation.....	AIPSCS	17
TTXPosition.....	AIPSDOMAIN	16
TTXWorldPosition.....	AIPSGEOGRAPHY	9

Bolt Beranek and Newman Inc.

Report No. 4752

Source File: AIPS

2. SOURCE FILE: AIPS

(FILECREATED " 8-Nov-80 22:49:37" <NEWAIPS>AIPS..3 686

changes to: AIPSUTILITYFNS SendMessage AIPSVARS AIPSCOMS

previous date: " 7-Nov-80 21:02:56" <NEWAIPS>AIPS..4)

(PRETTYCOMPRINT AIPSCOMS)

(RPAQQ AIPSCOMS ((FILES (COMPILED FROM NEWAIPS)

AIPSCS AIPSDEVICES AIPSDISPLAY AIPSDOMAIN

AIPSGEOGRAPHY AIPSMAP AIPSNAVAL AIPSNTDS

AIPSREALIZATION AIPSTABLE AIPSTEMP AIPSTODRAW

AIPSTS AIPSUTILITY AIPSWINDOW)))

(FILESLOAD (COMPILED FROM NEWAIPS)

AIPSCS AIPSDEVICES AIPSDISPLAY AIPSDOMAIN AIPSGEOGRAPHY AIPSMAP

AIPSNAVAL AIPSNTDS AIPSREALIZATION AIPSTABLE AIPSTEMP AIPSTODRAW

AIPSTS AIPSUTILITY AIPSWINDOW)

(DECLARE: DONTCOPY

(FILEMAP (NIL)))

STOP

Bolt Beranek and Newman Inc.

Report No. 4752

Source File: AIPSCS

3. SOURCE FILE: AIPSCS

Init2DCOORDINATESYSTEM.....	1
Init2DLINEARTRANSFIX.....	2
Init2DLINEARTRANSFORM.....	3
InitCARTESIANSYSTEM.....	4
InitCOORDINATESYSTEM.....	5
InitCoordinateSystemConcepts.....	6
InitINTEGERSYSTEM.....	7
InitLINEARTRANSFORM.....	8
InitMAPPING.....	9
InitORTHOGONALSYSTEM.....	10
InitTRANSFIX.....	11
InitTRANSLATION.....	12
InitVIEWSYSTEM.....	13
TT2DLinearTransFix.....	14
TT2DLinearTransform.....	15
TTTransFix.....	16
TTTranslation.....	17

(FILECREATED " 3-Jul-80 17:03:28" <NEWAIPS>AIPSCS..10 9183

changes to: AIPSCSCOMS

previous date: " 3-Jul-80 13:50:27" <NEWAIPS>AIPSCS..9)

(PRETTYCOMPRINT AIPSCSCOMS)

(RPAQQ AIPSCSCOMS ((FNS * AIPSCSINITFNS)
 (FNS * AIPSCSTOTRANSFORMFNS)
 (ADDVARS (CKLONEFILES AIPSCS))))

(RPAQQ AIPSCSINITFNS (Init2DCOORDINATESYSTEM Init2DLINEARTRANSFIX
 Init2DLINEARTRANSFORM
 InitCARTESIANSYSTEM
 InitCOORDINATESYSTEM
 InitCoordinateSystemConcepts
 InitINTEGERSYSTEM
 InitLINEARTRANSFORM
 InitMAPPING
 InitORTHOGONALSYSTEM
 InitTRANSFIX InitTRANSLATION
 InitVIEWSYSTEM))

(DEFINEQ

{1}

(Init2DCOORDINATESYSTEM
 [LAMBDA NIL

(* Edited by J.Gibbons on
 2-Jul-80.)

[concept 2DCOORDINATESYSTEM
 (specializes COORDINATESYSTEM)
 [roleaset NIL
 (mods Dimensionality@COORDINATESYSTEM)
 (vr 2)]]])

{2}

(Init2DLINEARTRANSFIX
 [LAMBDA NIL

(* Edited by J.Gibbons on
 3-Jul-80.)

(* Two dimensional, linear transforms from cartesian
 coordinate systems to integral cartesian coordinate systems)

[concept 2DLINEARTRANSFIX

```
(specializes 2DLINEARTRANSFORM)
[roleset NIL
 (mods To@2DLINEARTRANSFORM)
 (vr VIEWSYSTEM)]
(itags (ToTransform TT2DLinearTransFix)))]])
```

{3}

```
(Init2DLINEARTRANSFORM
 [LAMBDA NIL
```

```
(* Edited by J.Gibbons on
 3-Jul-80.)
```

```
[concept 2DLINEARTRANSFORM
 (specializes LINEARTRANSFORM)
 [roleset NIL
  (mods From@LINEARTRANSFORM)
  (vr CARTESIANSYSTEM)]
 [roleset NIL
  (mods To@LINEARTRANSFORM)
  (vr CARTESIANSYSTEM)]
 (itags (ToTransform TT2DLinearTransform)))]])
```

{4}

```
(InitCARTESIANSYSTEM
 [LAMBDA NIL
```

```
(* Edited by J.Gibbons on
 2-Jul-80.)
```

```
[concept CARTESIANSYSTEM
 (specializes 2DCOORDINATESYSTEM ORTHOGONALSYSTEM)]])
```

{5}

```
(InitCOORDINATESYSTEM
 [LAMBDA NIL
```

```
(* Edited by J.Gibbons on
 2-Jul-80.)
```

```
[concept COORDINATESYSTEM
 [roleset Name
  (vr STRINGP)]
```

(* This Role must be explicitly included because it is necessary for distinguishing among Coordinate Systems who do not as yet have Entries and Exits. It is not desirable to use a Concept Name as other than an indexing mechanism at the implementational level.)

```
[roleset Exit
  (number (0 NIL))
  (vr MAPPING)]
```

(* These are the transforms for
leaving the Coordinate System.)

```
[roleset Entry
  (number (0 NIL))
  (vr MAPPING)]
```

(* Transforms for entering the
Coordinate System.)

```
[roleset Dimensionality
  (modality Obligatory)
  (vr NUMBERP)]]]
```

{6}

```
(InitCoordinateSystemConcepts
[LAMBDA NIL
```

(* Edited by J.Gibbons on
3-Jul-80.)

(* Initializes those concepts having to do with Coordinate
Systems and mappings between them. Some major changes from the
previous paradigm occur among these concepts.
For example, positions are now simply LISTP, and must be
interpreted relative to some Coordinate System found in the
context of the description referencing the position.)

```
(InitCOORDINATESYSTEM)
(InitINTEGERSYSTEM)
(Init2DCOORDINATESYSTEM)
(InitORTHOGONALSYSTEM)
(InitCARTESIANSYSTEM)
(InitVIEWSYSTEM)
(InitMAPPING)
(InitLINEARTRANSFORM)
(Init2DLINEARTRANSFORM)
(Init2DLINEARTRANSFIX)
(InitTRANSLATION)
(InitTRANSFIX])
```

{7}

```
(InitINTEGERSYSTEM
[LAMBDA NIL
```

(* Edited by J.Gibbons on
2-Jul-80.)

```
[concept INTEGERSYSTEM
  (specializes COORDINATESYSTEM)
```

(* The purpose of this distinction is to signal that certain transformations must involve rounding.
Bit maps are integer systems.)

)))

{8}

(InitLINEARTRANSFORM
[LAMBDA NIL

(* Edited by J.Gibbons on
2-Jul-80.)

[concept LINEARTRANSFORM
(specializes MAPPING)
[roleset NIL
(mods From@MAPPING)
(vr ORTHOGONALSYSTEM)]
[roleset NIL
(mods To@MAPPING)
(vr ORTHOGONALSYSTEM)]
[roleset Origin
(vr LISTP)]
[roleset UnitVector
(vr LISTP)]]])

{9}

(InitMAPPING
[LAMBDA NIL

(* Edited by J.Gibbons on
2-Jul-80.)

[concept MAPPING
[roleset From
(modality Obligatory)
(number (1 NIL))
(vr COORDINATESYSTEM)]
[roleset To
(modality Obligatory)
(number (1 NIL))
(vr COORDINATESYSTEM)]

(* Should have an ITag whose referent is a function that can
be applied to a list of positions and smash them with
transformed coordinates.)

)))

{10}

```
(InitORTHOGONALSYSTEM
[LAMBDA NIL
```

```
(* Edited by J.Gibbons on
2-Jul-80.)
```

```
[concept ORTHOGONALSYSTEM
```

```
(* This Concept should have an SD that describes the
constraint of Orthogonality on the system's axes.)
```

```
(specializes COORDINATESYSTEM)]])
```

```
{11}
```

```
(InitTRANSFIX
[LAMBDA NIL
```

```
(* Edited by J.Gibbons on
3-Jul-80.)
(* Translation from cartesian
coordinate systems to integral
cartesian coordinate systems.)
```

```
[concept TRANSFIX
(specializes 2DLINEARTRANSFIX TRANSLATION)
(itags (ToTransform TTTransFix)))]])
```

```
{12}
```

```
(InitTRANSLATION
[LAMBDA NIL
```

```
(* Edited by J.Gibbons on
3-Jul-80.)
```

```
[concept TRANSLATION
```

```
(* Most of our transforms will
be purely translations.)
```

```
(specializes 2DLINEARTRANSFORM)
[roleset NIL
(mods UnitVector@2DLINEARTRANSFORM)
(vr <1 1>)]
(itags (ToTransform TTTranslation)))]])
```

```
{13}
```

```
(InitVIEWSYSTEM
[LAMBDA NIL
```

```
(* Edited by J.Gibbons on
2-Jul-80.)
```

```
[concept VIEWSYSTEM
```

```
(* This kind of coordinate
system can be used to describe
display regions.)
```

```

    (specializes INTEGERSYSTEM CARTESIANSYSTEM))])
)
(RPAQQ AIPSCSTOTTRANSFORMFNS (TT2DLinearTransFix TT2DLinearTransform
                              TTTransFix TTTranslation))
(DEFINEQ

```

{14}

```

(TT2DLinearTransFix
 [DLAMBDA ((transform IndividualConcept (SATISFIES (transform df
                                                    ^2DLINEARTRANSFIX)))
           (point (LISTP OF NUMBERP)))
 (CLISP: MIXED)
 (* Edited by J.Gibbons on
 3-Jul-80.)

 (* Performs a 2DLinearTransFixing -
 a 2DLinearTransformation to integer coordinates -
 on point according to transform. Returns the fixed, 2D
 linearly transformed point.)

```

```

(PROG [(unitVector (OR {transform;UnitVector}:1 '(1 1)))
       (origin (OR {transform;Origin}:1 '(0 0)
 (RETURN <(ROUND point:1*unitVector:1+origin:1)
          (ROUND point:2*unitVector:2+origin:2)
          >))])

```

{15}

```

(TT2DLinearTransform
 [DLAMBDA ((transform IndividualConcept (SATISFIES (transform df
                                                    ^2DLINEARTRANSFORM)))
           (point (LISTP OF NUMBERP)))
 (CLISP: MIXED)
 (* Edited by J.Gibbons on
 3-Jul-80.)

 (* Performs a 2DLinearTransformation on point according to
 transform. Returns the 2D linearly transformed point.)

```

```

(PROG [(unitVector (OR {transform;UnitVector}:1 '(1 1)))
       (origin (OR {transform;Origin}:1 '(0 0)
 (RETURN <point:1*unitVector:1+origin:1
          point:2*unitVector:2+origin:2>))])

```

{16}


```

(TTTransFix
[DLAMBDA ((transform IndividualConcept (SATISFIES (transform df `TRANSFIX)
))
          (point (LISTP OF NUMBERP)))
  (CLISP: MIXED)
  (* Edited by J.Gibbons on
    3-Jul-80.)

  (* Performs a TransFixing -
    a Translation to integer coordinates -
    on point according to transform. Returns the fixed, translated
    point.)

  (PROG [(origin (OR {transform;Origin}:1 '(0 0)
    (RETURN <(ROUND point:1+origin:1)
              (ROUND point:2+origin:2)
              >))])

```

{17}

```

(TTTranslation
[DLAMBDA ((transform IndividualConcept (SATISFIES (transform df
`TRANSLATION)))
          (point (LISTP OF NUMBERP)))
  (CLISP: MIXED)
  (* Edited by J.Gibbons on
    3-Jul-80.)
  (* Performs a Translation on
    point according to transform.
    Returns the translated point.)

  (PROG ((origin {transform;Origin}:1))
    (RETURN (if origin
      then <point:1+origin:1 point:2+origin:2>
      else point))))
)

(ADDTTOVAR CKLONEFILES AIPSCS)
STOP

```

4. SOURCE FILE: AIPSDEVICES

InitCONRACSCREEN.....1
InitCTYWINDOW.....2
InitDeviceConcepts...3
InitDevices.....4
InitHIDDENPLANE.....5
InitTURTLESCREEN.....6

(FILECREATED "19-Jan-81 01:23:51" <NEWAIPS>AIPSDEVICES..20 3933

changes to: InitDevices

previous date: " 3-Jan-81 01:05:32" <NEWAIPS>AIPSDEVICES..19)

(PRETTYCOMPRINT AIPSDEVICESCOMS)

(RPAQQ AIPSDEVICESCOMS ((FNS * AIPSDEVICESINITFNS)
 (GLOBALVARS TDRRegion)
 (ADDVARS (CKLONEFILES AIPSDEVICES))))

(RPAQQ AIPSDEVICESINITFNS (InitCONRACSCREEN InitCTYWINDOW
 InitDeviceConcepts InitDevices
 InitHIDDENPLANE
 InitTURTLESCREEN))

(DEFINEQ

{1}

(InitCONRACSCREEN
 [LAMBDA NIL

(* Edited by J.Gibbons on
 31-Dec-80.)

(* This is the IC for the PLANESURFACE which maps to the BMG
 PLANE viewed on the CONRAC monitor.)

(iconcept CONRACSCREEN (individuates PLANESURFACE VISIBLESURFACE)
 (irole Plane (vr `CONRACSCREEN))
 (irole PlaneNumber (vr 1))
 (irole Boundary (vr (TMRectangle):1))
 (irole Background (vr 0))

{2}

(InitCTYWINDOW
 [LAMBDA NIL

(* Edited by J.Gibbons on
 3-Jan-81.)
 (* This TTYWindow supports the
 controlling terminal on the
 TurtleScreen.)

(PROG (window)
 (window_ (SendMessage `TTYWINDOW 'ToMake `TURTLESCREEN):1)
 (KLZChangeConceptName window 'CTYWINDOW)
 (SendMessage window 'ToSize {`TURTLESCREEN;&Height}:1

```

                                {`TURTLESCREEN;&width}:1)
    (SendMessage window 'ToLocate '(0 0))

```

{3}

```

(InitDeviceConcepts
 [LAMBDA NIL

```

```

(* Edited by J.Gibbons on
 9-Dec-80.)

```

```

(* Initializes the concepts associated with the BMG devices.
These include the three ICs representing the PLANESURFACES of
bit mapped memory. They also include the IC for the initial
TTYWINDOW on the TURTLE screen called CTYWINDOW.
InitDevices must be called to realize these concepts.)

```

```

(InitTURTLESCREEN)
(InitCONRACSCREEN)
(InitHIDDENPLANE])

```

{4}

```

(InitDevices
 [LAMBDA NIL

```

```

(* Edited by J.Gibbons on
19-Jan-81.)

```

```

(* Initializes the BMG package and fleshes out the relevant
PlanesSurfaces. Also creates and draws the Controlling
TTYWindow on the TURTLESCREEN.)

```

```

(if (ASKUSER NIL 'N "Can I merely Reset (as opposed to Init) BMG? " NIL
    T)='Y
    then (BMGReset)
    else (BMGInit))
(SendMessage `CONRACSCREEN 'ToSize)
(SendMessage `CONRACSCREEN 'ToLocate)
(SendMessage `TURTLESCREEN 'ToSize)
(SendMessage `TURTLESCREEN 'ToLocate)
(SendMessage `HIDDENPLANE 'ToSize)
(SendMessage `HIDDENPLANE 'ToLocate)
(InitCTYWINDOW)
(SendMessage `CONRACSCREEN 'ToDraw)
(SendMessage `TURTLESCREEN 'ToDraw)
(SendMessage `HIDDENPLANE 'ToDraw)
(SendMessage `CTYWINDOW 'ToDraw])

```

{5}

```
(InitHIDDENPLANE
[LAMBDA NIL
```

```
(* Edited by J.Gibbons on
31-Dec-80.)
```

```
(* This is the IC for the PLANESURFACE which maps to the large
BMG PLANE. This PLANE is not viewed on any monitor.)
```

```
(iconcept HIDDENPLANE (individuates PLANESURFACE INVISIBLESURFACE)
  (irole Plane (vr `HIDDENPLANE))
  (irole PlaneNumber (vr 2))
  (irole Boundary (vr (TMRectangle):1))
  (irole Background (vr 0))
```

```
{6}
```

```
(InitTURTLESCREEN
[LAMBDA NIL
```

```
(* Edited by J.Gibbons on
31-Dec-80.)
```

```
(* This is the IC for the PLANESURFACE which maps to the BMG
PLANE viewed on the TURTLE monitor.)
```

```
(iconcept TURTLESCREEN (individuates PLANESURFACE VISIBLESURFACE)
  (irole Plane (vr `TURTLESCREEN))
  (irole PlaneNumber (vr 0))
  (irole Boundary (vr (TMRectangle):1))
  (irole Background (vr 0))
```

```
)
(DECLARE: DOEVAL@COMPILE DONTCOPY
```

```
(ADDTOVAR GLOBALVARS TDRegion)
)
```

```
(ADDTOVAR CKLONEFILES AIPSDEVICES)
STOP
```

5. SOURCE FILE: AIPSDISPLAY

AIPSGetSpecifiedMetaDescriptions.....	12
CollectTemplatesDescribingEntity.....	13
FindApplicableApplicationSlots.....	14
FindApplicationObjects.....	15
FindCostOfDisplay.....	16
FindCoveringIntendedApplication.....	17
FindTemplateObjects.....	18
FindTemplateSlots.....	19
GuageTemplateRGCoverageOfTemplate.....	20
InitDISPLAY.....	3
InitDISPLAYABSTRACTION.....	4
InitITEMPLATE.....	5
InitPRESENTATION.....	2
InitPresentatationModel.....	1
InitTEMPLATE.....	6
InitTEXT.....	7
MakeColumnString.....	21
MakeDefaultTextString.....	8
MakeDisplayName.....	9
MakeTextHeight.....	10
MakeTextWidth.....	11
SelectDefaultDisplay.....	22
SelectDisplay.....	23
SelectFromAmongSufficientDisplays.....	24
SupplyConceptMetaIndicator.....	25
SupplyRoleMetaIndicator.....	26
TemplateCGCoversTemplateCGP.....	27
TemplateOutSpecifiesTemplateP.....	28
TLDisplay.....	29

(FILECREATED "19-Jan-81 22:43:26" <NEWAIPS>AIPSDISPLAY..47 30099

changes to: InitDISPLAY

previous date: "19-Jan-81 01:02:42" <NEWAIPS>AIPSDISPLAY..46)

(PRETTYCOMPRINT AIPSDISPLAYCOMS)

```
(RPAQQ AIPSDISPLAYCOMS ((FNS * AIPSDISPLAYINITFNS)
                        (FNS * AIPSDISPLAYDERIVATIONFNS)
                        (FNS * AIPSDISPLAYUTILITYFNS)
                        (FNS * AIPSDISPLAYTOLOCATEFNS)
                        (DECLARE: EVAL@COMPILEWHEN
                          (NOT (BOUNDP (QUOTE BMGCOMS)))
                          DONTCOPY
                          (FILES <FONTWORK>BMGRECORDSANDVARS..0))
                        (ADDVARS (CKLONEFILES AIPSDISPLAY))))
```

```
(RPAQQ AIPSDISPLAYINITFNS (InitPresentationModel InitPRESENTATION
                                                                InitDISPLAY
                                                                InitDISPLAYABSTRACTION
                                                                InitITEMPLATE InitITEMPLATE
                                                                InitTEXT))
```

(DEFINEQ

{1}

(InitPresentationModel
[LAMBDA NIL

(* Edited by F.Zdybel on
22-Jul-80.)

(* Initializes the Presentation Model for AIPS.
These concepts describe the linkage between information being
output and the structure of the output.)

```
(InitPRESENTATION)
(InitDISPLAY)
(InitDISPLAYABSTRACTION)
(InitITEMPLATE)
(InitITEMPLATE)
(InitTEXT)
```

(* These initialization functions build SI-Net descriptions of
the principal Display types, such as Map, Table, Menu, etc.)


```
(InitTableConcepts)
(InitMapConcepts])
```

{2}

```
(InitPRESENTATION
[LAMBDA NIL
```

```
(* Edited by F.Zdybel on
4-Jul-80.)
```

```
[concept PRESENTATION
```

(* The top-level concept of the Presentation Model for AIPS. This description embraces all forms of information output, including synthesized natural language messages and synthesized speech, as well as graphic displays.)

```
[roleset Application
```

(* This Roleset is intended to express the binding of some particular (Individual) Presentation to a set of information which is to be displayed. Other Roles of the generic description of the Presentation may imply that further information may be involved in all Presentations of a particular type.)

```
(number (0 NIL))
(vr ITEMPLATE)]
[roleset Realization
```

(* This Roleset characterizes the internal structure of the Presentation as a group of constituents, which might be anything. This Role is differentiated and modified by generic descriptions descendant from Presentation.)

```
(number (0 NIL))
(vr THING)]])
```

{3}

```
(InitDISPLAY
[LAMBDA NIL
```

```
(* Edited by J.Gibbons on
19-Jan-81.)
```

```
[concept DISPLAY
```

(* A Graphically realized
Presentation.)

```
(specializes PRESENTATION DISPLAYITEM)
[roleset NIL
  (mods Realization@PRESENTATION)
  (vr DISPLAYITEM)]
```

(* A Display is either a Presentation or a Display Form.
Expansion of the description of any individual Display
terminates at the level of Display Forms
(the Viewing Organization and Realization Models of AIPS.))

```
]
[roleset Ground
  (modality Obligatory)
  (vr VIEWSURFACE)]
```

(* This Roleset describes the coordinate system and any
boundaries of the bit map surface on which the Display will be
realized.)

```
[roleset Name
  (vr STRINGP)
  (derivation (MakeDisplayName))]
(itags (ToDraw TDDisplay)
  (ToLocate TLDisplay))]]]
```

{4}

```
(InitDISPLAYABSTRACTION
[LAMBDA NIL
```

(* Edited by F.Zdybel on
14-Jul-80.)

```
[concept DISPLAYABSTRACTION
```

(* Abstractions of Displays are
individuals of this concept.)

```
(specializes DISPLAYITEMABSTRACTION)
[roleset IntendedApplication
  (number (1 NIL))
  (vr TEMPLATE)]
```

(* This role is used in the abstraction of a Display to
indicate what kind of information the Display can depict.
In the case of MAP, for example, the role is filled at the
abstraction of map by two templates that indicate that a map
can be used to depict the locations of physical objects or the
boundaries of regions.)

```
[roleset InformationRequirement
  (number (1 NIL))
  (vr TEMPLATE)]
```

(* Give a means of characterizing how much information will have to be known in order to realize a Display.)

]])

{5}

```
(InitITEMPLATE
 [LAMBDA NIL
```

(* Edited by F.Zdybel on
4-Jul-80.)

```
[concept ITEMPLATE
```

(* A Template whose Concept Group Role restricts it to the description of slots inherited by Individual Concepts.)

```
(specializes TEMPLATE)
[roleset NIL
  (mods ConceptGroup@TEMPLATE)
  (vr ICONCEPT)]]])
```

{6}

```
(InitTEMPLATE
 [LAMBDA NIL
```

(* Edited by F.Zdybel on
4-Jul-80.)

```
[concept TEMPLATE
```

(* A Template is a meta-description that characterizes a set of inherited Rolesets in an SI-Net hierarchy. Its' function is to indicate a chunk of information, as for example in characterizing the Domain World extension that is involved in an individual of Presentation.)

```
[roleset ConceptGroup
```

(* All Templates are individual descriptions.
The satisfiers of the Concept Group Role meta-indicate

Concepts in the Domain World as a way of specifying sub-lattices of the Domain Model within which inheritance of the specified Rolesets (specified by the Role Group Role of Template) is to be considered as part of the set of information being described.)

```
(number (0 NIL))
(vr CONCEPT)]
[roleset RoleGroup
```

(* The satisfiers of the Role Group Role meta-indicate Rolesets in the Domain World Model. Because of the inheritance of Rolesets, this is an intensional characterization of a set of inherited Roles. This characterization is limited in scope by the fillers of the Concept Group Role, which specify subsets of the SI-Net within which inheritance of the specified Roles is to be considered as generating the extension.)

```
(modality Obligatory)
(number (1 NIL))
(vr ROLE)]])
```

{7}

```
(InitTEXT
[LAMBDA NIL
```

```
(* Edited by J.Gibbons on
12-Dec-80.)
```

```
[concept TEXT
```

(* Instances of Text may be called on to display practically anything. There is therefore no IntendedApplication Template on TEXT's abstraction. Instead, TEXT is selected by default whenever no other Display with specified IntendedApplication can be discovered that discharges the purpose at hand.)

```
(specializes DISPLAY)
(abstraction [iconcept TEXTABSTR of DISPLAYABSTRACTION])
[roleset NIL
(mods Realization@DISPLAY)
(number 0)]
```

(* A kluge of sorts, TEXT itself has no Realization, since it is itself also a descendant of DISPLAYITEM.)

```

[roleset String
  (modality Obligatory)
  (number 1)
  (vr STRINGP)
  (fillwhenmade)
  [derivation (MakeDefaultTextString (Prerequisites Application)
                                     (Arguments (CAR
                                                  {$$CONCEPT;Application})))]

[roleset Font
  (vr SMALLP)
  (defaultfiller BMGDefaultRegionFontNumber)]
[roleset LineSpacing
  (vr SMALLP)
  (defaultfiller BMGDefaultRegionLineSpacing)]
[roleset NIL
  (mods Height@DISPLAY)
  (derivation (MakeTextHeight (Prerequisites String Font LineSpacing)
                              (Arguments {$$CONCEPT;String}:1
                                          {$$CONCEPT;Font}:1
                                          {$$CONCEPT;LineSpacing}:1))))

[roleset NIL
  (mods Width@DISPLAY)
  (derivation (MakeTextWidth (Prerequisites String Font)
                             (Arguments {$$CONCEPT;String}:1
                                          {$$CONCEPT;Font}:1))))

(itags (ToDraw TDText))]]]
)

(RPAQQ AIPSDISPLAYDERIVATIONFNS (MakeDefaultTextString MakeDisplayName
                                                         MakeTextHeight
                                                         MakeTextWidth))

(DEFINEQ

{8}

(MakeDefaultTextString
  [DLAMBDA ((application IndividualConcept (SATISFIES
                                             (application df `ITEMPLATE
                                             and
                                             (FLENGTH
                                              {application;ConceptGroup})=1
                                             and
                                             (FLENGTH
                                              {application;RoleGroup})=1)))

    (RETURNS (LST OF STRINGP)))

    (* Edited by F.Zdybel on

```

9-Dec-80.)

(* Makes the best possible effort to display application as a piece of text, incorporating Concept and Role Names where necessary. The RoleGroup of application is expected to consist of a single role metaindicator, the ConceptGroup of a single iconcept metaindicator.)

```
(for roleFiller in (KLFindRoleValues (KLGetMetaDescribedEntity
                                     {application;ConceptGroup}:1
                                     (KLGetNamedContext
                                      'AIPSDOMAINMETA))
                  (KLGetMetaDescribedEntity
                   {application;RoleGroup}:1
                   (KLGetNamedContext
                    'AIPSDOMAINMETA)))
  collect (if (KLConceptP roleFiller)
             then (CAR (NLSETQ (SendMessage roleFiller 'ToTextify)))
              or (MKSTRING (OR {roleFiller;Name}:1
                              (KLGetConceptName roleFiller)))
             elseif roleFiller = T
              then "Yes"
             elseif roleFiller = NIL
              then "No"
             else (MKSTRING roleFiller))
  finally (RETURN <(if ~$$VAL:1
                      then $$VAL:1
                      else (MakeColumnString $$VAL))
           >))])
```

{9}

```
(MakeDisplayName
[DLAMBDA ((RETURNS (LST OF STRINGP)))
```

```
(* Edited by F.Zdybel on
17-Nov-80.)
```

(* The last ditch effort to create a name for a Map is based on the name of the prototype map and a GENSYM.)

```
(PROG ((currentNumber ((KLGetData $$PROTOTYPE 'NumberCreatedSoFar):1
                        or 0)+ 1))
      (KLAttachDatum $$PROTOTYPE 'NumberCreatedSoFar currentNumber)
      (RETURN <(CONCAT (KLGetConceptName $$PROTOTYPE)
                      " #" currentNumber)
              >))])
```

{10}

```

(MakeTextHeight
  [DLAMBDA ((string STRINGP)
            (fontNumber BMGFontNumber)
            (lineSpacing SMALLP))

    (* Edited by J.Gibbons on
       12-Dec-80.)

    (* Calculates the Height of a TTYString in pixels according to
       the specified Font. NOTE: This also depends on the LineSpacing
       of the BMGRegion in which the TTYString will appear.
       We cannot handle this aspect currently and use the LineSpacing
       value of the current Graphics Region.)

    <(if string equal ""
        then 0
        else lineSpacing*(bind characterCode for character
                           in (UNPACK string) eachtime characterCode_ (
                               CHCON1 character)
                           count characterCode=10 or characterCode=31)+(
                               BMGDescribeFont fontNumber):LoadedFont\FontRecord:Font\Height)
    >])

```

{11}

```

(MakeTextWidth
  [DLAMBDA ((string STRINGP)
            (fontNumber BMGFontNumber))

    (* Edited by J.Gibbons on
       15-Dec-80.)

    (* Calculates the maximum width of the given String in pixels,
       ie. the width of its longest line, according to the specified
       font.)

    (bind characterCode currentWidth_0
      maximumWidth_0 for character in (UNPACK string)
      do (characterCode_ (CHCON1 character))
        (if characterCode=13 or characterCode=31
          then maximumWidth_ (IMAX currentWidth maximumWidth)
              currentWidth_0
          elseif characterCode~=10
            then (currentWidth_ currentWidth+(GetFontCharacterDef
                                                fontNumber
                                                characterCode)
                :CharDef\Width))
      finally (RETURN <(IMAX currentWidth maximumWidth)

```

```

>)))]
)
(RPAQQ AIPSDISPLAYUTILITYFNS (AIPSGetSpecifiedMetaDescriptions
    CollectTemplatesDescribingEntity
    FindApplicableApplicationSlots
    FindApplicationObjects
    FindCostOfDisplay
    FindCoveringIntendedApplication
    FindTemplateObjects
    FindTemplateSlots
    GuageTemplateRGCoverageOfTemplate
    MakeColumnString
    SelectDefaultDisplay
    SelectDisplay
    SelectFromAmongSufficientDisplays
    SupplyConceptMetaIndicator
    SupplyRoleMetaIndicator
    TemplateCGCoversTemplateCGP
    TemplateOutSpecifiesTemplateP))
(DEFINEQ

```

{12}

```

(AIPSGetSpecifiedMetaDescriptions
[DLAMBDA ((describedEntity MetaDescribableEntity)
  (metaDescRestriction Concept)
  (RETURNS (LST OF IndividualConcept)))
  (* Edited by F.Zdybel on
  19-Jun-80.)

```

```

(* Finds all metadescriptions of describedEntity that are eq
to or subCs of metaDescRestriction)

```

```

(for metaDescription in (KLGetMetaDescriptions describedEntity)
  when (metaDescription=metaDescRestriction)
    or (metaDescription df metaDescRestriction)
  collect metaDescription))

```

{13}

```

(CollectTemplatesDescribingEntity
[DLAMBDA ((domainEntity IndividualConcept)
  (templateList (LST OF IndividualConcept) (SATISFIES
    (
HomogenousConceptListP templateList `TEMPLATE)))
  (RETURNS (LST OF IndividualConcept) (SATISFIES
    (HomogenousConceptListP

```



```

VALUE `TEMPLATE)))
(* Edited by F.Zdybel on
30-Jul-80.)

```

```

(* Searches through the templates in templateList and collects
together all templates that refer to the domainEntity.)

```

```

(for template in templateList
  when (for entityMetaDescription in {template;ConceptGroup}
    bind templateEntity eachtime templateEntity_ (
      KLGetMetaDescribedEntity
      entityMetaDescription
      (KLGetNamedContext
        'AIPSDOMAINMETA))

    thereis domainEntity=templateEntity
      or (KLGenericConceptP templateEntity)
      and domainEntity df templateEntity)
  and [for roleMetaDescription in {template;RoleGroup}
    thereis (KLFindRoleValues domainEntity
      (KLGetMetaDescribedEntity
        roleMetaDescription
        (KLGetNamedContext
          'AIPSDOMAINMETA]

  collect template)])

```

{14}

```

(FindApplicableApplicationSlots
  [DLAMBDA ((domainEntity IndividualConcept)
    (templateList (LST OF IndividualConcept) (SATISFIES
      (
        HomogenousConceptListP templateList `TEMPLATE)))
    (RETURNS (LST OF Role)))

```

```

(* Edited by F.Zdybel on
5-Aug-80.)

```

```

(* Searches through the templates on templateList and collects
together all of the roles metaindicated by templates that CG
cover domainEntity.)

```

```

(for template in templateList
  when (for templateObject in (FindTemplateObjects template)
    thereis (if (KLIndividualConceptP templateObject)
      then domainEntity=templateObject
      else domainEntity df templateObject))
  join (FindTemplateSlots template) finally (RETURN
    (INTERSECTION $SVAL
      $SVAL))))

```

{15}

```

(FindApplicationObjects
[DLAMBDA ((templateList (LST OF IndividualConcept) (SATISFIES
HomogenousConceptListP templateList `TEMPLATE)))
  (RETURNS (LST OF Concept)))
(* Edited by F.Zdybel on
5-Aug-80.)

(* Goes through templateList and constructs a list of all the
domain entities indicated by the ConceptGroup fillers of the
templates.)

(for template in templateList join (FindTemplateObjects template)
  finally (RETURN (INTERSECTION $$VAL $$VAL))))

```

{16}

```

(FindCostOfDisplay
[DLAMBDA ((display GenericConcept (SATISFIES (display df `DISPLAY)))
  (proposedApplication IndividualConcept (SATISFIES
    (proposedApplication
      df `ITEMPLATE)))
  (RETURNS NUMBERP))
(* Edited by F.Zdybel on
4-Jul-80.)

(* At the moment, the best measure we can get for the cost of
a display is the number of RoleGroup Role fillers of the
InformationRequirement that applies to the subjects
(ConceptGroup Role fillers) of proposedApplication.
The assumption is that the InformationRequirement templates on
a display's abstraction are disjoint with respect to subject.)

```

```

(FLENGTH {(for costTemplate in {(KLGetAbstraction display)
  ;InformationRequirement}
  thereis (TemplateCGCoversTemplateCGP costTemplate
    proposedApplication))
  ;RoleGroup}}))

```

{17}

```

(FindCoveringIntendedApplication
[DLAMBDA ((display GenericConcept (SATISFIES (KLIsConceptDescendantP

```

```

                                display `DISPLAY)))
    (proposedApplication IndividualConcept (SATISFIES
                                           (KLIsConceptDescendantP
                                            proposedApplication
                                            `ITEMPLATE)))
    (RETURNS (ONEOF (IndividualConcept (SATISFIES
                                         (KLIsConceptDescendantP
                                          VALUE `TEMPLATE)))
                   NIL)))

```

(* Edited by F.Zdybel on
23-Jul-80.)

(* Finds the IntendedApplication template on display's
abstraction that has to do with the subjects
(ConceptGroup fillers) of proposedApplication.
The assumption is that IntendedApplications on a display's
abstraction are disjoint with respect to subject.)

```

(for intendedApplication in {(KLGetAbstraction display)
                             ;IntendedApplication}
  thereis (TemplateCGCoversTemplateCGP intendedApplication
        proposedApplication)))

```

{18}

```

(FindTemplateObjects
 [DLAMBDA ((template IndividualConcept (SATISFIES (KLIsConceptDescendantP
                                                  template `TEMPLATE)))
          (RETURNS (LST OF Concept)))

```

(* Edited by F.Zdybel on
30-Jul-80.)

```

(for objectDescriptor in {template;ConceptGroup}
  collect (KLGetMetaDescribedEntity objectDescriptor
        (KLGetNamedContext 'AIPSDOMAINMETA)
        )))

```

{19}

```

(FindTemplateSlots
 [DLAMBDA ((template IndividualConcept (SATISFIES (template df `TEMPLATE)))
          (RETURNS (LST OF Role)))

```

(* Edited by F.Zdybel on
5-Aug-80.)

```

(for slotDescriptor in {template;RoleGroup}
  collect (KLGetMetaDescribedEntity slotDescriptor
        (KLGetNamedContext 'AIPSDOMAINMETA)
        )))

```

{ 20 }

```
(GuageTemplateRGCoverageOfTemplate
[DLAMBDA ((superTemplate IndividualConcept (SATISFIES (
KLIsConceptDescendantP superTemplate `TEMPLATE)))
(subTemplate IndividualConcept (SATISFIES (
KLIsConceptDescendantP subTemplate `TEMPLATE)))
(RETURNS FIXP))
```

(* Edited by F.Zdybel on
30-Jul-80.)

(* Finds the number of RoleGroup fillers in superTemplate that
metaindicate Roles that are ancestors of
(or EQ to) the roles metaindicated by the RoleGroup fillers of
subTemplate.)

```
(for superRoleMetaIndicator in {superTemplate;RoleGroup}
  bind (context_(KLGetNamedContext 'AIPSDOMAINMETA))
        degreeOfCoverage_0
        superRole subRoles
  first subRoles_(for subRoleMetaIndicator in {subTemplate;RoleGroup}
                  collect (KLGetMetaDescribedEntity
                           subRoleMetaIndicator
                           context))
  eachtime superRole_(KLGetMetaDescribedEntity superRoleMetaIndicator
                                                  context)
  when (for subRole in subRoles thereis (KLIsRoleDescendantP subRole
                                                             superRole))
  do degreeOfCoverage_degreeOfCoverage+1
  finally (RETURN degreeOfCoverage)))
```

{ 21 }

```
(MakeColumnString
[LAMBDA (stringList)
```

(* Edited by F.Zdybel on
9-Dec-80.)

(* Takes a list of strings and returns a single string which
includes n-1 CRLF's among the n elements of the input list.)

```
(APPLY 'CONCAT (for fillerString in stringList::1
                 bind (firstElt _ stringList:1)
                 join <"
" fillerString> finally (RETURN <firstElt ! $$VAL>)))
```

{ 22 }

```
(SelectDefaultDisplay
  [DLAMBDA ((proposedApplication IndividualConcept (SATISFIES
                                                    (proposedApplication
                                                       df `ITEMPLATE)))
            (RETURNS GenericConcept (SATISFIES (VALUE df `DISPLAY))))
    (* Edited by F.Zdybel on
      20-Jul-80.)
```

(* Selects from either TABLE or TEXT depending on whether there is more than one RoleGroup filler in proposedApplication. This is a last ditch effort to find a way of displaying some information for which no tailored Display exists. Note that a Table or a Text can be arbitrarily combined into the containing Display even if neither TABLE or TEXT are value restrictions on that Display's Realization Role.)

```
(if {proposedApplication;RoleGroup}::1
    then `TABLE
    else `TEXT))
```

{23}

```
(SelectDisplay
  [DLAMBDA ((proposedApplication IndividualConcept (SATISFIES
                                                    (KLIsConceptDescendantP
                                                       proposedApplication
                                                       `ITEMPLATE)))
            (possiblePrototypes (LST OF GenericConcept) (SATISFIES
                                                            (
HomogenousConceptListP possiblePrototypes `DISPLAY)))
            (RETURNS GenericConcept (SATISFIES (KLIsConceptDescendantP
                                                  VALUE `DISPLAY))))
    (* Edited by F.Zdybel on
      4-Aug-80.)
```

(* Searches through possiblePrototypes to find the prototype MapItem that best suits applicationTemplate. At this point the general approach is to find the set of all displays with sufficient IntendedApplication to support proposedApplication (or to find the one display that most supports proposedApplication). If more than one display is applicable, we will (for the moment) take the most specific one. Eventually, the algorithm should go on to consider the cost of the candidate displays.)

```
[for prototype in possiblePrototypes
```

```

bind (maximumCoverage_ (FLENGTH {proposedApplication;RoleGroup}))
    currentBestPrototype currentBestDegreeOfCoverage_0
    prototypeIntendedApplicationOfInterest
    candidateDegreeOfCoverage completelySufficientPrototypes
when prototypeIntendedApplicationOfInterest_ (
    FindCoveringIntendedApplication prototype proposedApplication)
do (if candidateDegreeOfCoverage_ (GuageTemplateRGCoverageOfTemplate
    prototypeIntendedApplicationOfInterest proposedApplication)
    =maximumCoverage
    then completelySufficientPrototypes_
        < !! completelySufficientPrototypes prototype>
    elseif candidateDegreeOfCoverage gt currentBestDegreeOfCoverage
    then currentBestPrototype_ prototype
        currentBestDegreeOfCoverage candidateDegreeOfCoverage)
finally (RETURN (if completelySufficientPrototypes
    then (SelectFromAmongSufficientDisplays
        completelySufficientPrototypes
        proposedApplication)
    elseif currentBestPrototype
    then currentBestPrototype
    else (SelectDefaultDisplay proposedApplication)))

```

{ 24 }

```

(SelectFromAmongSufficientDisplays
[DLAMBDA ((alternativeDisplays (LST OF GenericConcept) (SATISFIES
HomogenousConceptListP alternativeDisplays `DISPLAY)))
    (proposedApplication IndividualConcept (SATISFIES
        (proposedApplication
        df `ITEMPLATE)))
    (RETURNS GenericConcept (SATISFIES (VALUE df `DISPLAY))))
    (* Edited by F.Zdybel on
    18-Jul-80.)

```

(* Tries to find the display of minimum information cost.
If several candidates have the same cost, tries to find the
most specialized.)

```

[for prototypeDisplay in alternativeDisplays bind mostMinimalPrototypes
costOfMostMinimalPrototypes _ MAXFIX)
    costOfCandidate
    eachtime costOfCandidate_ (FindCostOfDisplay prototypeDisplay
        proposedApplication)

```

(* We are assuming for the moment that the
InformationRequirements role is filled by at most one
template.)

```

do (if costOfCandidate lt costOfMostMinimalPrototypes
    then mostMinimalPrototypes_ <
        prototypeDisplay>
    elseif costOfCandidate = costOfMostMinimalPrototypes
    then mostMinimalPrototypes_ <!! mostMinimalPrototypes
        prototypeDisplay>)
finally (RETURN (if ~mostMinimalPrototypes:1
    then mostMinimalPrototypes:1
    else (FindMostSpecializedConcept
        mostMinimalPrototypes)))

```

{25}

```

(SupplyConceptMetaIndicator
  [DLAMBDA ((concept Concept)
    (contextName LITATOM)
    (RETURNS IndividualConcept))

```

(* Edited by F.Zdybel on
11-Jul-80.)

(* Finds or creates a metadescription for the given concept in
either the given context or the first of the currently active
contexts.)

```

[OR (CAR (KLGetMetaDescriptions concept (if contextName
    then (KLGetNamedContext
        contextName)
    or
        (KLCreateContext
        contextName)
    else $$CONTEXTS:1)))
  [iconcept of CONCEPT
    [contexts ((if contextName
        then (KLGetNamedContext contextName)
        or (KLCreateContext contextName)
        else $$CONTEXTS:1))
      (metadescribes (atomval concept))]]])

```

{26}

```

(SupplyRoleMetaIndicator
  [DLAMBDA ((role Role)
    (contextName LITATOM)
    (RETURNS IndividualConcept))

```

(* Edited by F.Zdybel on

11-Jul-80.)

(* Either finds or creates a meta-description of the given role in either the given context or the first of the currently active contexts.)

```
[OR (CAR (KLGetMetaDescriptions role (if contextName
                                     then (KLGetNamedContext
                                           contextName)
                                     or (KLCreateContext
                                           contextName)
                                     else $$CONTEXTS:1)))
  [iconcept of ROLE
    [contexts ((if contextName
                  then (KLGetNamedContext contextName)
                  or (KLCreateContext contextName)
                  else $$CONTEXTS:1))
      (metadescribes (atomval role))]]]]]
```

{ 27 }

```
(TemplateCGCoversTemplateCGP
[DLAMBDA ((superTemplate IndividualConcept (SATISFIES (superTemplate
                                                       df `TEMPLATE)))
  (subTemplate IndividualConcept (SATISFIES (subTemplate df
                                                       `TEMPLATE)))
  (RETURNS [ONEOF NIL (IndividualConcept (SATISFIES (VALUE df
                                                       `TEMPLATE))])
    (* Edited by F.Zdybel on
      4-Jul-80.)
  (PROG ((superObjects (FindTemplateObjects superTemplate))
    (subObjects (FindTemplateObjects subTemplate)))
    (RETURN (if (for subObject in subObjects
                  always (for superObject in superObjects
                        thereis subObject df superObject))
      then superTemplate
      else NIL)))))
```

{ 28 }

```
(TemplateOutSpecifiesTemplateP
[DLAMBDA ((subTemplate IndividualConcept (SATISFIES (subTemplate df
                                                       `TEMPLATE)))
  (superTemplate IndividualConcept (SATISFIES (superTemplate
                                                       df `TEMPLATE)))
  (RETURNS [ONEOF NIL (IndividualConcept (SATISFIES (VALUE df
                                                       `TEMPLATE))])
    (* Edited by F.Zdybel on
```


27-Jun-80.)

(* Tests to see whether the subTemplate names only objects that are descended from the objects named in the superTemplate, and that all of the slots named in the superTemplate are also included in the subTemplate. Put another way, this function checks to be sure that the ConceptGroup of superTemplate covers the ConceptGroup of subTemplate and the RoleGroup of subTemplate covers the RoleGroup of superTemplate.)

```
(PROG ((superObjects (FindTemplateObjects superTemplate))
      (superSlots (FindTemplateSlots superTemplate))
      (subObjects (FindTemplateObjects subTemplate))
      (subSlots (FindTemplateSlots subTemplate)))
  (RETURN (if (for subObject in subObjects
                always (for superObject in superObjects
                        thereis subObject df superObject))
              and (for superSlot in superSlots
                    always (for subSlot in subSlots
                            thereis (KLIsRoleDescendantP
                                     subSlot superSlot)))
              then subTemplate))))))

(RPAQQ AIPSDISPLAYTOLOCATEFNS (TLDisplay))
(DEFINEQ
```

{ 29 }

```
(TLDisplay
 [DLAMBDA ((displayDescr IndividualConcept (SATISFIES (displayDescr df
                                                         ^DISPLAY)))
           (displayLoc (ONEOF NIL (LISTP OF FIXP (SATISFIES ^displayLoc::2)
                                                         )))
           (displayGround [ONEOF NIL (IndividualConcept
                                         (SATISFIES (displayGround df
                                                         ^VIEWSURFACE]))
                           (* Edited by F.Zdybel on
                              16-Dec-80.)
```

(* If given a ground, changes or establishes the Ground Role of displayDescr. If given a location, changes or establishes the Location Role of displayDescr.)

```
(PROG (currentLocationRole currentGroundRole)
  (if displayLoc
    then (if currentLocationRole_ (KLFindOneNamedInstanceRole
```

```
        displayDescr 'Location)
      then (KLChangeRoleValue currentLocationRole
            displayLoc)
      else (KLSatisfyRole Location@DISPLAY displayDescr
            <displayLoc>)))
  (if displayGround
    then (if currentGroundRole_ (KLFindOneNamedInstanceRole
                                displayDescr
                                'Ground)
      then (KLChangeRoleValue currentGroundRole
                                displayGround)
      else (KLSatisfyRole Ground@DISPLAY displayDescr
                                <displayGround>]))
)
(DECLARE: EVAL@COMPILEWHEN (NOT (BOUNDP (QUOTE BMGCOMS)))) DONTCOPY
(FILESLoad <FONTWORK>BMGRECORDSANDVARS..0)
)

(ADDTovar CKLONEFILES AIPSDISPLAY)
STOP
```

6. SOURCE FILE: AIPSDOMAIN

InitCommonDomainModel.....1
InitDISTANCE.....2
InitDISTANCEUNIT.....3
InitINTERVAL.....4
InitINTERVALUNIT.....5
InitMETRIC.....6
InitPHYSOBJECT.....7
InitPLACE.....8
InitPOSITION.....9
InitREGION.....10
InitSPEED.....11
InitSPEEDUNIT.....12
InitTIME.....13
InitUNIT.....14
InitVEHICLE.....15
TTXPosition.....16

(FILECREATED "10-Dec-80 23:41:31" <NEWAIPS>AIPSDOMAIN..5 8837

changes to: TTXPosition

previous date: "10-Dec-80 01:48:00" <NEWAIPS>AIPSDOMAIN..4)

(PRETTYCOMPRINT AIPSDOMAINCOMS)

(RPAQQ AIPSDOMAINCOMS ((FNS * AIPSDOMAININITFNS)
 (FNS * AIPSDOMAINTOTEXTIFYFNS)
 (ADDVARS (CKLONEFILES AIPSDOMAIN))))

(RPAQQ AIPSDOMAININITFNS (InitCommonDomainModel InitDISTANCE
 InitDISTANCEUNIT
 InitINTERVAL
 InitINTERVALUNIT InitMETRIC
 InitPHYSOBJECT InitPLACE
 InitPOSITION InitREGION
 InitSPEED InitSPEEDUNIT
 InitTIME InitUNIT
 InitVEHICLE))

(DEFINEQ

{1}

(InitCommonDomainModel
 [LAMBDA NIL

(* Edited by F.Zdybel on
 10-Jul-80.)

(* Sets up some of the very high level concepts of the domain
 world, which are referenced by templates associated with
 various Displays.)

(InitPHYSOBJECT)
 (InitVEHICLE)
 (InitPLACE)
 (InitPOSITION)
 (InitREGION)
 (InitTIME)
 (InitMETRIC)
 (InitDISTANCE)
 (InitINTERVAL)
 (InitSPEED)
 (InitUNIT)
 (InitDISTANCEUNIT)
 (InitINTERVALUNIT)
 (InitSPEEDUNIT))

{2}

```
(InitDISTANCE
  [LAMBDA NIL
```

```
    [concept DISTANCE
```

```
      (specializes METRIC)
      [roleset NIL
        (mods Unit@METRIC)
        (vr DISTANCEUNIT)]]])
```

```
(* Edited by F.Zdybel on
9-Jul-80.)
```

```
(* Describes distance, linking
a distance unit with a
magnitude.)
```

{3}

```
(InitDISTANCEUNIT
  [LAMBDA NIL
```

```
    [concept DISTANCEUNIT
```

```
(* Should establish a way of expressing equivalences among
distance units. For the moment merely establishes the
abbreviation for the unit.)
```

```
      (specializes UNIT)
      [roleset NIL
        (mods Subject@UNIT)
        (vr DISTANCE)]]])
```

```
(* Edited by F.Zdybel on
9-Jul-80.)
```

{4}

```
(InitINTERVAL
  [LAMBDA NIL
```

```
    [concept INTERVAL
```

```
      (specializes METRIC)
      [roleset NIL
        (mods Unit@METRIC)
        (vr INTERVALUNIT)]]])
```

```
(* Edited by F.Zdybel on
9-Jul-80.)
```

```
(* The distance between two
points in time.)
```

{5}

```
(InitINTERVALUNIT
[LAMBDA NIL
 [concept INTERVALUNIT
```

(* Describes any unit used for
measuring the passage of time.)

```
    (specializes UNIT)
  [roleset NIL
    (mods Subject@UNIT)
    (vr INTERVAL)]]])
```

{6}

```
(InitMETRIC
[LAMBDA NIL
 [concept METRIC
```

(* Describes measurement of
something.)

```
    [roleset Magnitude
      (vr NUMBERP)]
  [roleset Unit
    (vr UNIT)]]])
```

{7}

```
(InitPHYSOBJECT
[LAMBDA NIL
 [concept PHYSOBJECT
```

(* Edited by F.Zdybel on
9-Jul-80.)

(* Something with form,
substance, structure and
locability.)

```
    [roleset Position
      (vr PLACE)]
```

(* For the moment, this is all that concerns us: objects are
located somewhere in the domain world.)

```
    [roleset Name
      (vr STRINGP)]
```

(* Most anything might have a
name.))])

{8}

```
(InitPLACE
[LAMBDA NIL
```

(* Edited by F.Zdybel on

9-Jul-80.)

[concept PLACE

(* This might be either a specific position or a region
 (it also might be several other things like "over Paris"
 "4:00 high" etc, but for the moment let us confine ourselves
 ...))

[roleset Constraint
 (vr THING)]

(* Might be a Point location (LISTP), or a curve, or the
 boundary of a Region. For the moment we will eschew attempting
 to characterize these things at this value restriction.)

[roleset Continuum
 (vr COORDINATESYSTEM)]

(* The constraint will be
 relative to this coordinate
 system.))])

{9}

(InitPOSITION
 [LAMBDA NIL

(* Edited by F.Zdybel on
 9-Dec-80.)

[concept POSITION
 (specializes PLACE)

(* A discrete position in some
 coordinate system.)

[roleset Location
 (mods Constraint@PLACE)
 (vr LISTP)]

(* Again, a simple list is used
 for the coordinate tuple.)

(itags (ToTextify TTXPosition))))

{10}

(InitREGION
 [LAMBDA NIL

(* Edited by J.Gibbons on
 11-Nov-80.)

[concept REGION
 (specializes PLACE)

(* A Multi-purpose Concept. ITags on this concept should refer to functions that can decide whether a position is "inside" the region and can compute the distance to a boundary of the region from a given position.)

```
[roleset Boundary
  (diffs Constraint@PLACE)
  (vr CLOSEDCURVE)]
```

(* A Region never has more than a single boundary.)

```
[roleset Aperture
  (diffs Constraint@PLACE)
  (number (0 NIL))
  (vr REGION)]
```

(* A Region may have holes in it.)

```
[roleset Height
  (vr NUMBERP)]
[roleset Width
  (vr NUMBERP)]]])
```

{11}

```
(InitSPEED
[LAMBDA NIL
```

(* Edited by F.Zdybel on 9-Jul-80.)

```
[concept SPEED
```

(* Describes speed, linking a speed unit with a magnitude.)

```
(specializes METRIC)
[roleset NIL
  (mods Unit@METRIC)
  (vr SPEEDUNIT)]]])
```

{12}

```
(InitSPEEDUNIT
[LAMBDA NIL
```

(* Edited by F.Zdybel on 9-Jul-80.)

```
[concept SPEEDUNIT
```

(* Links a unit for measuring speed to units for measuring distance and time.)

```
(specializes UNIT)
[roleset DistanceUnit
```



```

      (vr DISTANCEUNIT)]
    [roleset IntervalUnit
      (vr INTERVALUNIT)]
    [roleset NIL
      (mods Subject@UNIT)
      (vr SPEED)]]])

```

{13}

```

(InitTIME
  [LAMBDA NIL
    [concept TIME

```

(* Like PLACE, TIME is a more or less specific constraint imposed on a CoordinateSystem. Eons and Instants are both Times.)

```

[roleset Constraint
  (vr THING)]

```

(* This VR should become more elegant when we have time for elegance. Generally, the filler of this role is a NUMBERP.)

```

[roleset Continuum
  (vr TIMESYSTEM)]

```

(* And there is (eventually) surely more than one TimeSystem (e.g.: Zulu time, GMT, Sidereal Time, Stardate, etc.))

```

  ]])

```

{14}

```

(InitUNIT
  [LAMBDA NIL
    [concept UNIT

```

(* Edited by F.Zdybel on
9-Jul-80.)

(* A unit of measurment of some metric
(e.g., speed, time, distance, temperature.))

```

[roleset Subject

```

```
(vr METRIC)))]])
```

{15}

```
(InitVEHICLE
[LAMBDA NIL
```

```
(* Edited by F.Zdybel on
13-Jul-80.)
```

```
[concept VEHICLE
```

```
(* A physical object capable of moving around under its own
power and carrying something.)
```

```
(specializes PHYSOBJECT)
[roleset Medium
(vr ATOM)]
```

```
(* Expected to be something on
the order of SubSurface, Air or
Surface.)
```

```
[roleset Course
(vr NUMBERP)]
```

```
(* Expected to be a number between 0 and 360, indicating the
direction of motion of the Vehicle.)
```

```
[roleset Speed
(vr SPEED)]
```

```
(* Gives the speed of the vehicle in terms of distance units
per time unit.)
```

```
]])
)
```

```
(RPAQQ AIPSDOMAINTEXTIFYFNS (TTXPosition))
(DEFINEQ
```

{16}

```
(TTXPosition
[LAMBDA (positionDescr)
```

```
(* Edited by F.Zdybel on
10-Dec-80.)
```

```
(* Attempts to construct a string which states the position
described by positionDescr. At this level, this is merely a
```

matter of formatting with some parens and a comma.)

```
(PROG ((coordinateList (CAR {positionDescr;Location})))
  (RETURN (APPLY 'CONCAT (for coordinate in coordinateList::l
    bind (firstElt coordinateList:l)
    join "<", " coordinate>"
    finally (RETURN
      "<(" (MKSTRING firstElt)
      " ! < !! $$VAL )" ">>])
  )
(ADDTOTVAR CKLONEFILES AIPSDOMAIN)
STOP
```


7. SOURCE FILE: AIPSGEOGRAPHY

InitGeographyModel...2
InitGEOREGION.....1
InitLANDFEATURE.....3
InitLANDMASS.....4
InitNATION.....5
InitWATERBODY.....6
InitWATERFEATURE.....7
InitWORLDPOSITION....8
RptCoord.....10
TTXWorldPosition.....9

(FILECREATED "11-Dec-80 00:01:03" <NEWAIPS>AIPSGEOGRAPHY..9 6330

changes to: AIPSGEOGRAPHYCOMS AIPSGEOGRAPHYUTILITYFNS TTXWorldPosition
previous date: "10-Dec-80 22:03:20" <NEWAIPS>AIPSGEOGRAPHY..7)

(PRETTYCOMPRINT AIPSGEOGRAPHYCOMS)

(RPAQQ AIPSGEOGRAPHYCOMS ((FNS * AIPSGEOGRAPHYINITFNS)
 (FNS * AIPSGEOGRAPHYTOTEXTIFYFNS)
 (FNS * AIPSGEOGRAPHYUTILITYFNS)
 (ADVARS (CKLONEFILES AIPSGEOGRAPHY))))

(RPAQQ AIPSGEOGRAPHYINITFNS (InitGEOREGION InitGeographyModel
 InitLANDFEATURE InitLANDMASS
 InitNATION InitWATERBODY
 InitWATERFEATURE
 InitWORLDPOSITION))

(DEFINEQ

{1}

(InitGEOREGION
[LAMBDA NIL

(* Edited by F.Zdybel on
10-Dec-80.)

[concept GEOREGION

(* A region on surface of the earth that is modelled as having
a bounded area. This might describe cites, for example, but
not villages. It might describe some rivers, but not others.)

(specializes REGION GEOFEATURE)
[roleset Name
 (vr STRINGP)]
[roleset NIL
 (mods Continuum@REGION)
 (vr [iconcept WORLDCARTESIAN])]

(* Notice that we are assuming for the moment a single world
coordinate system that is not spherical.)

[roleset Feature

(number (0 NIL))
(vr GEOFEATURE)

(* Notice that a feature may also be a region, so this allows
us to cope with islands in the sea, cities on the land, etc.)

]]])

{2}

(InitGeographyModel
[LAMBDA NIL

(* Edited by F.Zdybel on
10-Dec-80.)

(* This initialization function grows the descriptions for
Geography Model. This portion of the Domain Model is generally
necessary for complete description of a certain class of Maps
which include outlines of land masses, bodies of water, etc.)

(InitWORLDPOSITION)
(InitNATION)
(InitGEOREGION)
(InitWATERBODY)
(InitLANDMASS)
(InitWATERFEATURE)
(InitLANDFEATURE])

{3}

(InitLANDFEATURE
[LAMBDA NIL

(* Edited by F.Zdybel on
19-Jun-80.)

[concept LANDFEATURE
(specializes GEOFEATURE)
(subcs RIVER MOUNTAIN CITY PENINSULA ISTHMUS)]]])

{4}

(InitLANDMASS
[LAMBDA NIL

(* Edited by F.Zdybel on
19-Jun-80.)

[concept LANDMASS
(specializes GEOREGION GEOFEATURE)
(subcs CONTINENT ISLAND CITY)
[role set Water
(diffs Feature@GEOREGION)

```
(vr WATERBODY)]
[roleset NIL
 (diffs Feature@GEOREGION)
 (vr LANDFEATURE)]]])
```

{5}

```
(InitNATION
[LAMBDA NIL
```

```
(* Edited by F.Zdybel on
19-Jun-80.)
```

```
[concept NATION
```

(* One of the top level concepts of the Geography Model. Further expansion of this Concept will make possible specified depiction of national attributes such as Capital, Seaport, IndustrialCenter, MilitaryInstallation, etc. The Geography model is actually a small Domain World in and of itself.)

```
[roleset Name
 (vr STRINGP)]
[roleset Territory
 (vr GEOREGION)]]])
```

{6}

```
(InitWATERBODY
[LAMBDA NIL
```

```
(* Edited by F.Zdybel on
19-Jun-80.)
```

```
[concept WATERBODY
```

```
(* One of two general classes
of Geo Regions ...)
```

```
(specializes GEOREGION GEOFEATURE)
(subcs OCEAN SEA LAKE)
[roleset Land
 (diffs Feature@GEOREGION)
 (vr LANDMASS)
```

```
(* The Other of the two general
classes ...)]
```

```
[roleset NIL
 (diffs Feature@GEOREGION)
 (vr WATERFEATURE)]
```

(* Note that a Water Body is allowed to contain land masses but not land features. Thus a city on an island in the sea is not necessarily retrieved as a Feature of the sea.)

```
]])
```


{7}

```
(InitWATERFEATURE
  [LAMBDA NIL
```

```
(* Edited by F.Zdybel on
  19-Jun-80.)
```

```
  [concept WATERFEATURE
    (specializes GEOFEATURE)
    (subcs REEF SEA STRAIGHT CHANNEL GULF)]])
```

{8}

```
(InitWORLDPOSITION
  [LAMBDA NIL
```

```
(* Edited by F.Zdybel on
  10-Dec-80.)
```

```
  [concept WORLDPOSITION
```

```
    (* A position expressed in world coordinates
      (hopefully soon to be actually a spherical system, but
      currently fudged as rectangular radians.))
```

```
      (specializes POSITION)
      [roleset NIL
        (mods Continuum@POSITION)
        (vr [iconcept WORLDCARTESIAN of CARTESIANSYSTEM])]
      (itags (ToTextify TTXWorldPosition))])
```

```
)
(RPAQQ AIPSGEOGRAPHYTOTEXTIFYFNS (TTXWorldPosition))
(DEFINEQ
```

{9}

```
(TTXWorldPosition
  [LAMBDA (positionDescr)
```

```
(* Edited by F.Zdybel on
  11-Dec-80.)
```

```
  (* Takes spherical coordinates expressed in radians and outputs
    a string phrased in terms of Latitude and Longitude.)
```

```
  (PROG ((coordinateList (CAR {positionDescr;Location})))
    (RETURN (APPLY 'CONCAT < !! (RptCoord coordinateList:1 T)
```

```
    " ! (RptCoord coordinateList:2)
```

```
>])
```

)

```
(RPAQQ AIPSGEOGRAPHYUTILITYFNS (RptCoord))
(DEFINEQ
```

{10}

```
(RptCoord
[LAMBDA (radianNum latFlg)
  (CLISP: FAST FLOATING)
```

```
(* Edited by F.Zdybel on
10-Dec-80.)
```

```
(* Function turns a latitude or longitude expressed as
floating point radians into a list of the form degrees minutes
seconds and direction, with appropriate descriptive labels for
the fields.)
```

```
(PROG (degrees minutes seconds token)
  (degrees_radianNum*180.0/3.141593)
  (minutes_(degrees-(degrees_(FIX degrees)))*60.0)
  (seconds_(FIX (minutes-(minutes_(FIX minutes)))*6000.0)/100.0)
  (token_(if radianNum gt 0.0
    then (if latFlg
      then 'N.
      else 'E.)
    elseif latFlg
      then 'S.
      else 'W.))
  (RETURN <(if latFlg
    then "Lat. "
    else "Lng. ")
    (ABS degrees)
    "d "
    (ABS minutes)
    "' "
    (ABS seconds)
    "% " token>))
```

)

```
(ADDTTOVAR CKLONEFILES AIPSGEOGRAPHY)
STOP
```

8. SOURCE FILE: AIPSMAP

EstablishDomainEntityPositionInCS.....	23
FindDomainCoordinateSystem.....	8
FindMapEntitiesFromApplication.....	9
FindMapItemPrototypes.....	21
FindMappingBetweenCSs.....	22
FindScaleUnitDistance.....	10
InitMAP.....	1
InitMapConcepts.....	6
InitMAPGRID.....	2
InitMAPITEM.....	3
InitMAPLEGEND.....	4
InitMAPWITHBACKGROUND.....	5
InitRECTANGULARMAP.....	7
MakeGroundForMapWithBackground.....	11
MakeLegendApplication.....	12
MakeLinearMapTransformAndGroundFromEntities.....	13
MakeLinearMapTransformFromGroundAndEntities.....	14
MakeMapItemApplication.....	24
MakeMapItemLabel?.....	15
MakeMapItems.....	16
MakeMapLabel.....	17
MakeMapTransformForMapWithBackground.....	18
MakeRectangularMapBorder.....	19
MakeSimpleMapLegend.....	20
MapPositionDescription.....	25

(FILECREATED "13-Nov-80 03:55:19" <NEWAIPS>AIPSMAP..57 40193

changes to: MakeLegendApplication MakeSimpleMapLegend

previous date: "13-Nov-80 01:24:02" <NEWAIPS>AIPSMAP..56)

(PRETTYCOMPRINT AIPSMAPCOMS)

(RPAQQ AIPSMAPCOMS ((FNS * AIPSMAPINITFNS)
 (FNS * AIPSMAPDERIVATIONFNS)
 (FNS * AIPSMAPUTILITYFNS)
 (ADDVARS (CKLONEFILES AIPSMAP))))

(RPAQQ AIPSMAPINITFNS (InitMAP InitMAPGRID InitMAPITEM InitMAPLEGEND
 InitMAPWITHBACKGROUND InitMapConcepts
 InitRECTANGULARMAP))

(DEFINEQ

{1}

(InitMAP
 [LAMBDA NIL

(* Edited by F.Zdybel on
 29-Oct-80.)

[concept MAP

(* Map is a major Presentation
 form.)

(specializes DISPLAY)
 (abstraction [iconcept MAPABSTR of DISPLAYABSTRACTION])
 [roleset EntityList
 (vr LISTP)
 (derivation (FindMapEntitiesFromApplication (Prerequisites
 Application)
 (Arguments
 {\$\$CONCEPT;Application})))
 (fillwhenmade)]

(* The EntityList Role give a place to determine the set of objects with discrete physical locations that are to be included in the Map. This set is used to drive many of the other derivation procedures on Roles of MAP. For example, the functions used to derive the size of the Ground for the Map know how to make it big enough to include things with discrete locations, but not how to make it big enough to include arbitrarily shaped areas (in the future, hopefully, this can be fixed so that regions can also be included in the EntityList.) The EntityList Role also provides a place to resolve coersions of several symbols

into a single symbol for purposes of uncluttering
(e.g.: three ships become a task group.))

```
[roleset NIL
  (mods Name@DISPLAY)
  (modality Obligatory)
  (fillwhenmade)]
```

(* This is the string that gets made into a Label for the Map.
The name should be based on the largest georegion included in
the Map (if this can be determined) or on the Template Role of
the Map if it is simple enough to be summarized in a short
string. At least we will see something here of the form "MAP #"
<sequential Map number>.)

```
[roleset DomainCoordSys
  (vr 2DCOORDINATESYSTEM)
  [derivation (FindDomainCoordinateSystem
    (Prerequisites EntityList)
    (Arguments (CAR {$$CONCEPT;EntityList}))
    (KLFindVRsOfRole (KLFindOneNamedGenericRole
      $$CONCEPT
      'DomainCoordSys])
    (fillwhenmade)]]
```

(* This is the domain world CoordinateSystem that is assumed
to be shared by all of the items being depicted.
It is not the Coordinate System of the depictions.
Some of the depicted items may have to have their domain world
locations re-expressed in terms of this coordinate system.)

```
[roleset MapItemTransform
  (modality Obligatory)
  (vr MAPPING)]
[roleset Label
  (diffs Realization@DISPLAY)
  (vr TEXT)
  [derivation (MakeMapLabel (Prerequisites Name Ground)
    (Arguments (CAR {$$CONCEPT;Name}))
    (CAR {$$CONCEPT;Ground}))
    (fillwhenmade)]
[roleset Border
  (diffs Realization@DISPLAY)
  (vr CLOSEDCURVE)]
```

(* For some types of projections, the border of the Map is not necessarily a Rectangle, or even a Polygon.)

```
[roleset Legend
  (diffs Realization@DISPLAY)
  (vr MAPLEGEND)
  (derivation (MakeSimpleMapLegend (Prerequisites Name Item)
                                     (Arguments {CAR {$$CONCEPT;Name}}
                                                  {$$CONCEPT;Item})))
  (fillwhenmade)]
```

(* This may be one of the places where one desires to have a part of a Presentation viewed through one window while other parts of the Presentation are viewed through another (i.e. when scrolling over a Map with a Window one may wish that the Legend is always in view.))

```
[roleset ReferenceGrid
  (diffs Realization@DISPLAY)
  (vr MAPGRID)]
[roleset Item
  (modality Obligatory)
  (number (1 NIL))
  (diffs Realization@DISPLAY)
  (vr MAPITEM)
  (derivation (MakeMapItems (Prerequisites EntityList Application
                                     Ground)
                             (Arguments {$$CONCEPT;Application}{CAR
                                     {$$CONCEPT;EntityList}}
                                     {CAR {$$CONCEPT;Ground}}))
  (fillwhenmade)]
```

(* Those things having discrete locations in the domain world.)

```
(itags (MinInterItemSpacing 10)
       (MinItemSpread 150))
```

(* These factors are the minimum spacing (in pixels) between any two items on a Map, and the minimum permissible spread (in pixels) in any dimension among all the items, respectively.)

```
(tags (DefaultPrototype `RECTANGULARMAP)))
[iconcept MAPABSTR
```

(* The abstract individual representing the class of all maps.)

```

[irole IntendedApplication
  (vr [iconcept MAPIAPL1]))]
[irole IntendedApplication
  (vr [iconcept MAPIAPL2]))]
[iconcept MAPIAPL1

```

(* The first and obvious intended use of a Map is to show the location of something)

```

(individuates TEMPLATE)
[irole ConceptGroup
  (vr (SupplyConceptMetaIndicator [concept PHYSOBJECT]
    'AIPSDOMAINMETA)))]
[irole RoleGroup
  (vr (SupplyRoleMetaIndicator [roleset Position of PHYSOBJECT]
    'AIPSDOMAINMETA)))]
[iconcept MAPIAPL2

```

(* A second intended use of a Map is to show the boundaries of regions.)

```

(individuates TEMPLATE)
[irole ConceptGroup
  (vr (SupplyConceptMetaIndicator [concept REGION]
    'AIPSDOMAINMETA)))]
[irole RoleGroup
  (vr (SupplyRoleMetaIndicator [roleset Boundary of REGION]
    'AIPSDOMAINMETA)))]

```

{2}

```

(InitMAPGRID
  [LAMBDA NIL

```

(* Edited by F.Zdybel on
19-Jun-80.)

```

[concept MAPGRID
  (specializes DISPLAY)
  [roleset ReferenceLine
    (diffs Realization@DISPLAY)
    (modality Obligatory)
    (number (4 NIL))
    (vr CURVE)]
  [roleset ReferenceLabel
    (diffs Realization@DISPLAY)
    (modality Obligatory)
    (number (4 NIL))
    (vr TEXT)]

```

(* Attached procedures know how
to realize the particular types
of reference grids.)))]

{3}

(InitMAPITEM
[LAMBDA NIL

(* Edited by F.Zdybel on
29-Oct-80.)

[concept MAPITEM

(* A symbol in a Map, which denotes the location of some
locable domain world entity.)

```
(specializes DISPLAY)
(abstraction [iconcept MAPITEMABSTR of DISPLAYABSTRACTION])
[roleset NIL
  (mods Application@DISPLAY)]
[roleset Label
  (diffs Realization@DISPLAY)
  (vr TEXT)
  (derivation (MakeMapItemLabel? (Prerequisites Application Ground)
    (Arguments { $$CONCEPT;Application}:1
                { $$CONCEPT;Ground}:1)))
  (fillwhenmade)]]
[iconcept MAPITEMABSTR
  [irole IntendedApplication
    (vr [iconcept MAPITEMIAPL1])]
  [irole FixedComponent
    [vr (Make LINESEGMENT EndPoint '((-6 -6)
      (6 6))]
    [irole FixedComponent
      [vr (Make LINESEGMENT EndPoint '((-6 6)
        (6 -6))]
        (* X marks the spot ...)]
  [iconcept MAPITEMIAPL1
```

(* For the moment we will restrict ourselves to displaying the
discrete locations and names of physical objects.)

```
(individuates TEMPLATE)
[irole ConceptGroup
  (vr (SupplyConceptMetaIndicator [concept PHYSOBJECT]
    'AIPSDOMAINMETA))]
[irole RoleGroup
  (vr (SupplyRoleMetaIndicator [roleset Position of PHYSOBJECT]
```



```

'AIPSDOMAINMETA))]]
[irole RoleGroup
  (vr (SupplyRoleMetaIndicator [roleset Name of PHYSOBJECT]
    'AIPSDOMAINMETA))]]]]

```

{4}

```

(InitMAPLEGEND
[LAMBDA NIL

```

```

(* Edited by F.Zdybel on
29-Oct-80.)

```

```

[concept MAPLEGEND
  (specializes ARRANGEMENT)
  [roleset ScaleIkon
    (diffs Item@ARRANGEMENT)
    (vr MAPSCALE)
    (derivation (MakeLegendScaleIkonFromApplication (Prerequisites
      Application)
      (Arguments
        {$$CONCEPT;Application}))))]]

```

(* This is the funny little bar with the colored stripes and the numbers under it. Note that this part of the Legend is sensitive to the current scale of the map, which may change due to Window-driven scaling.)

```

[roleset Table
  (modality Obligatory)
  (diffs Item@ARRANGEMENT)
  (vr LEGENDTABLE)
  (derivation (MakeLegendTableFromApplication (Prerequisites
    Application)
    (Arguments
      {$$CONCEPT;Application}))))
  (fillwhenmade)]

```

(* This is the little correspondence Table between map symbols and explanatory text.)

```

[roleset Label
  (modality Obligatory)
  (diffs Item@ARRANGEMENT)
  (vr TEXT)
  (itags (DefaultFiller (Make TEXT String "LEGEND")))]
  (* Probably says something
  brilliant like "Legend".))]

```

{5}

```
(InitMAPWITHBACKGROUND
[LAMBDA NIL
```

```
(* Edited by F.Zdybel on
17-Jul-80.)
```

```
[concept MAPWITHBACKGROUND
```

```
(* Any kind of map that already has one or more display forms
on the FixedComponent Role of its abstraction
(for geographic regions and reference grid) and hence has a
fixed DomainCoordSys and very particular value restrictions
for the MapItemTransform and Ground Roles.
For example, the value restriction of the MapItemTransform
Role will be a Generic 2DLinearTransform whose Origin and
UnitVector roles are filled with particular LISTP
(even though the From and To Roles remain to be filled at the
various individuals of that Generic Concept.))
```

```
(specializes MAP)
```

```
(* There is no point in having an abstraction for this
particular type of Map, because we will generally always be
dealing with its generic descendants.)
```

```
[roleset NIL
(mods Ground@MAP)
(derivation (MakeGroundForMapWithBackground
              (Arguments (KLFindVRsOfRole (KLFindOneNamedGenericRole
                                           $$CONCEPT 'Ground)):1)))
]

[roleset NIL
(mods MapItemTransform@MAP)
(derivation (MakeMapTransformForMapWithBackground
              (Prerequisites Ground)
              (Arguments (KLFindVRsOfRole (KLFindOneNamedGenericRole
                                           $$CONCEPT
                                           'MapItemTransform)):1
                          { $$CONCEPT;DomainCoordSys}:1
                          { $$CONCEPT;Ground}:1))))]

[roleset NIL
(mods ReferenceGrid@MAP)
(number (0 0))]]]
```

```
{6}
```

```
(InitMapConcepts
[LAMBDA NIL
```

```
(* Edited by F.Zdybel on
```

17-Jul-80.)

(* Initializes that portion of the AIPS Presentation Model
having to do with Maps.)

```
(InitMAP)
(InitRECTANGULARMAP)
(InitMAPITEM)
(InitMAPLEGEND)
(InitMAPGRID)
(InitMAPWITHBACKGROUND))
```

{7}

```
(InitRECTANGULARMAP
 [LAMBDA NIL
```

(* Edited by F.Zdybel on
29-Oct-80.)

```
 [concept RECTANGULARMAP
```

(* This is a flat rectangular map, as opposed to some kind of
polar or conic projection with a funny shaped border and
non-constant values for the screen distance equivalent to a
given domain world distance.)

```
(specializes MAP)
[roleset NIL
 (mods DomainCoordSys@MAP)
 (vr CARTESIANSYSTEM)]
```

(* For the moment we will limit ourselves here to cartesian
domain coordinate systems)

```
[roleset NIL
 (mods MapItemTransform@MAP)
 (vr 2DLINEARTRANSFORM)
 [derivation (MakeLinearMapTransformFromGroundAndEntities
 (Prerequisites DomainCoordSys Ground EntityList)
 (Arguments (CAR {$$CONCEPT;EntityList})
 (CAR {$$CONCEPT;Ground})
 (CAR {$$CONCEPT;DomainCoordSys})))
 (MakeLinearMapTransformAndGroundFromEntities
 (Prerequisites DomainCoordSys EntityList)
 (Arguments (CAR {$$CONCEPT;EntityList})
 (CAR {$$CONCEPT;DomainCoordSys}]))]
 [roleset ScaleUnit
```

```
(vr DISTANCEUNIT)
(itags (DefaultFiller `MILE)))
```

(* The filler of this role should be selected according to the user's convenience. It is used in constructing the reference grid and the scale ikon of the legend.
For the moment, the filler of this role is simply defaulted to NM at NTDSMAP.)

```
[roleset ScaleDistance
  (vr NUMBERP)
  [derivation (FindScaleUnitDistance (Prerequisites DomainCoordSys
                                         Ground
                                         MapItemTransform)
                                         (Arguments (CAR
                                                       {$$CONCEPT;MapItemTransform})
                                                       (CAR
                                                       {$$CONCEPT;DomainCoordSys})
                                                       (CAR
                                                       {$$CONCEPT;Ground;Continuum}))])]
```

(* This is the distance equivalent to the filler of the Scale Unit Role in the Coordinate System of the Presentation.)

```
[roleset NIL
  (mods Border@MAP)
  (vr RECTANGLE)
  [derivation (MakeRectangularMapBorder (Prerequisites Ground)
                                         (Arguments (CAR
                                                       {$$CONCEPT;Ground;Boundary;LowerLeft})
                                                       (CAR
                                                       {$$CONCEPT;Ground;Boundary;UpperRight})
                                                       (CAR
                                                       {$$CONCEPT;Ground}))
                                         (fillwhenmade))]
  [roleset NIL
    (mods Ground@MAP)
    (derivation (Prerequisites MapItemTransform))]
  (itags (MapMarginFactor .07))
```

(* This factor is used to derive the amount by which the ground should be larger than the maximum spread in item X or Y coordinates.)

```
]])
)
```

```

(RPAQQ AIPSMAPDERIVATIONFNS (FindDomainCoordinateSystem
                             FindMapEntitiesFromApplication
                             FindScaleUnitDistance
                             MakeGroundForMapWithBackground
                             MakeLegendApplication
                             MakeLinearMapTransformAndGroundFromEntities
                             MakeLinearMapTransformFromGroundAndEntities
                             MakeMapItemLabel?
                             MakeMapItems
                             MakeMapLabel
                             MakeMapTransformForMapWithBackground
                             MakeRectangularMapBorder
                             MakeSimpleMapLegend))

```

```

(DEFINEQ

```

```

      {8}

```

```

(FindDomainCoordinateSystem
 [DLAMBDA ((domainEntityList (LST OF IndividualConcept))
          (coordSysRestrictions (LST OF GenericConcept)) (SATISFIES
          (
HomogenousConceptListP coordSysRestrictions `COORDINATESYSTEM)))
 (RETURNS (LST OF IndividualConcept) (SATISFIES (VALUE:1 df
          coordSysRestrictions:1))))
 (* Edited by F.Zdybel on
 13-Nov-80.)

```

(* Looks through the input entity list and finds the coordinate system that is a descendant of all of coordSysRestrictions and is used by the majority of the entries. If no coordinate system can be found associated with any of the entities that satisfies all of coordSysRestrictions, something more complicated should be done to find a suitable domain coordinate system and establish the necessary mappings. For the moment, however, we will content ourselves with calling ERROR.)

```

(for alternativeEntry
  in (for domainEntity in domainEntityList bind domainCoordSys
        domainCoordSysEntry
        coordinateSystemAlternatives
    do

```

(* Make an Alist to accumulate the different coordinate systems used by the various domain entities, where the value of each entry is the list of domain entities sharing the coordinate system.)

```

        (if domainCoordSysEntry (ASSOC domainCoordSys_
                                     {domainEntity;Position;Continuum}:1
                                     coordinateSystemAlternatives)
            then < !! domainCoordSysEntry domainEntity>
            else coordinateSystemAlternatives_
                < !! coordinateSystemAlternatives
                    <domainCoordSys domainEntity>>)
            finally (RETURN coordinateSystemAlternatives))
    bind currentCandidate (currentCandidatePopularity_0)
    alternative popularityOfAlternative
    when popularityOfAlternative (FLENGTH alternativeEntry) gt
        currentCandidatePopularity
        and (for restriction in coordSysRestrictions
            first alternative_alternativeEntry:1
            always alternative df restriction)
    do

```

(* Find the most used domain coordinate system that satisfies
all of the constraints.)

```

        (currentCandidate alternative)
        (currentCandidatePopularity_popularityOfAlternative)
    finally (if currentCandidate
        then (RETURN <currentCandidate>)
        else (ERROR
            "unable to establish acceptable domain coordinate system:
                $$CONCEPT DERIVENOBREAKFLG))))))

```

{9}

```

(FindMapEntitiesFromApplication
[DLAMBDA ((mapApplicationList (LST OF IndividualConcept) (SATISFIES
    (
HomogenousConceptListP mapApplicationList `ITEMPLATE)))
    (RETURNS (LST OF (LST OF IndividualConcept))))
    (* Edited by F.Zdybel on
    13-Nov-80.)

```

(* Examines mapApplicationList (the Application Role fillers
of Some Map) and determines all of the discretely locatable
physical objects that are described by these templates.
This list is helpful for a number of derivation functions
attached to Roles of MAP.)

```

(for template in mapApplicationList
    join (for domainObjectMetaDescription in {template;ConceptGroup}

```

```

bind domainObjectDescription
when domainObjectDescription (KLGetMetaDescribedEntity
    domainObjectMetaDescription
    (KLGetNamedContext 'AIPSDOMAINMETA))
    and {domainObjectDescription;Position;Constraint}:1
    df `LISTP
collect domainObjectDescription)
finally (RETURN <(INTERSECTION $$VAL $$VAL)
    >)))]

```

{10}

```

(FindScaleUnitDistance
[DLAMBDA ((scaleUnit IndividualConcept (SATISFIES (scaleUnit df
    `DISTANCEUNIT)))
    (domainCoordSys IndividualConcept (SATISFIES
    (domainCoordSys df
    `COORDINATESYSTEM)))
    (vsCoordSys IndividualConcept (SATISFIES (vsCoordSys df
    `CARTESIANSYSTEM)))
    (RETURNS (LST OF FLOATP)))
(* Edited by F.Zdybel on
13-Nov-80.)

```

(* Determines the distance represented by scaleUnit in domainCoordSys in terms of an increment along an axis of vsCoordSys. Otherwise, the result is returned as FLOATP. Apparently we will require to associate with each domain coordinate system an equivalence relationship between distance units (e.g.: one NM in the Lat/Long system is one minute of arc))

```

(ERROR "derivation function not implemented yet.
")))]

```

{11}

```

(MakeGroundForMapWithBackground
[DLAMBDA ((groundVR GenericConcept (SATISFIES (groundVR df `VIEWSURFACE)))
    (RETURNS (LST OF IndividualConcept) (SATISFIES (VALUE:1 df
    `VIEWSURFACE))))
(* Edited by F.Zdybel on
13-Nov-80.)

```

(* Creates an individual of groundVR, a Generic ViewSurface with Boundary Role already restricted to a particular ClosedCurve. Since this Value Description will already be inherited by the new individual, there should be no need to

copy it onto an IRole. Assumes that the function that derives the MapItemTransform will fill in the Exit role on the Continuum of the Ground.)

```
<(APPLY* 'Make (KLGetConceptName groundVR)
    'Continuum
    (APPLY* 'Make (KLGetConceptName {groundVR;Continuum}:1)))
>))
```

{12}

```
(MakeLegendApplication
[DLAMBDA ((itemRestrictionList (LST OF GenericConcept) (SATISFIES
    (
HomogenousConceptListP itemRestrictionList `MAPITEM)))
    (mapItemList (LST OF IndividualConcept) (SATISFIES
        (
HomogenousConceptListP mapItemList `MAPITEM)))
    (RETURNS (LST OF IndividualConcept) (SATISFIES VALUE:1 df
        `ITEMPLATE)))
    (* Edited by F.Zdybel on
    13-Nov-80.)
```

(* Looks through the items in the Map and devises an application role filler for the Legend of the Map.)

```
<(Make ITEMPLATE ConceptGroup
    (for itemType in (for itemRestriction in itemRestrictionList
        join <itemRestriction
            !(KLFindSubConcepts itemRestriction)
        >
        finally (RETURN (INTERSECTION $$VAL $$VAL)))
    when (for item in mapItemList thereis (KLZIsConceptDescendantP
        item itemType))
    join (for template in {(KLGetAbstraction itemType)
        ;IntendedApplication}
        collect (SupplyConceptMetaIndicator template
            'AIPSINTERNALMETA)))
    RoleGroup
    (SupplyRoleMetaIndicator ConceptGroup@TEMPLATE 'AIPSINTERNALMETA)
    )
>))
```

{13}

```
(MakeLinearMapTransformAndGroundFromEntities
[DLAMBDA ((domainEntityList (LST OF IndividualConcept))
```



```

(domainCoordinateSystem IndividualConcept (SATISFIES
                                          domainCoordinateSystem df
                                          ^CARTESIANSYSTEM))
(RETURNS (LST OF IndividualConcept) (SATISFIES VALUE:1 df
                                          ^2DLINEARTRANSFORM)))
(CLISP: MIXED (RECORD (XCoord YCoord)))
(* Edited by F.Zdybel on
13-Nov-80.)

(* Finds the minimum separation between map entities and
selects a transform such that this minimum distance just maps
onto some minimum desirable visual separation in terms of
pixels. Thus, since window scaling is always greater than 1,
this visual distance may become larger due to scaling
transform associated with a Window onto the VS, but it will
never become smaller. As a side effect, derives the Ground
role of $$CONCEPT, and establishes the new transform as the
Exit and Entry Role fillers of the domainCoordinateSystem and
the Continuum of the new ground, respectively.)

(for tail on (for entity in domainEntityList when Position@entity
              collect (CAR ((EstablishDomainEntityPositionInCS
                             entity domainCoordinateSystem)
                             ;Location))))))
bind location locationXCoord locationYCoord minX_MAX.FLOAT
minY_MAX.FLOAT
maxX_MIN.FLOAT
maxY_MIN.FLOAT
minXSpacing_MAX.FLOAT
minYSpacing_MAX.FLOAT
minSimpleDistance_MAX.FLOAT
minRealDistance scaleFactor mapGround itemsWidth itemsHeight
groundWidth groundHeight [minItemSpread
  (CAR (KLFindIData $$CONCEPT '(MinItemSpread)
    [mapMarginFactor (CAR (KLFindIData $$CONCEPT '(MapMarginFactor)
      xMargin yMargin xOrigin yOrigin transform
do (location tail:1)
  (locationXCoord_location:XCoord)
  (locationYCoord_location:YCoord)
  (minX (MIN locationXCoord minX))
  (maxX (MAX locationXCoord maxX))
  (minY (MIN locationYCoord minY))
  (maxY (MAX locationYCoord maxY))
  (for otherLocation in tail::1 bind currentXSpacing
                                     currentYSpacing
                                     currentSimpleDistance
  do (currentXSpacing (ABS locationXCoord-otherLocation:XCoord))
      (currentYSpacing (ABS locationYCoord-otherLocation:YCoord))
      (currentSimpleDistance_currentXSpacing+currentYSpacing)
      (if currentSimpleDistance lt minSimpleDistance

```

```

      then minSimpleDistance_currentSimpleDistance
           minXSpacing_currentXSpacing
           minYSpacing_currentYSpacing))
finally (minRealDistance (SQRT minXSpacing^2+minYSpacing^2))
      (scaleFactor_(MAX (CAR (KLFindIData $$CONCEPT '(
                               MinInterItemSpacing)))
                       /minRealDistance minItemSpread/(maxY-minY)
                       minItemSpread/(maxX-minX)))
      (itemsWidth (maxX-minX)*scaleFactor)
      (itemsHeight (maxY-minY)*scaleFactor)
      (groundWidth_itemsWidth+(2*(
           xMargin_itemsWidth*mapMarginFactor)))
      (groundHeight_itemsHeight+(2*(
           yMargin_itemsHeight*mapMarginFactor)))
      (mapGround_(Make VIEWSURFACE Continuum
                    (Make CARTESIANSYSTEM)
                    Boundary
                    (Make RECTANGLE UpperRight
                          <groundWidth groundHeight> LowerLeft
                          <0.0 0.0>)))
      (xOrigin_xMargin-scaleFactor*minX)
      (yOrigin_yMargin-scaleFactor*minY)
      (transform_(Make 2DLINEARTRANSFORM From
                      domainCoordinateSystem To
                      (CAR {mapGround;Continuum})
                      Origin <xOrigin yOrigin> UnitVector
                      <scaleFactor scaleFactor>))
      [irole Ground of (atomval $$CONCEPT)
        (vr (atomval mapGround))]
      [irole Exit of (atomval domainCoordinateSystem)
        (vr (atomval transform))]
      [irole Entry of (CAR {mapGround;Continuum})
        (vr (atomval transform))]
      (RETURN <transform>)))]

```

{14}

```

(MakeLinearMapTransformFromGroundAndEntities
[DLAMBDA ((domainEntityList (LST OF IndividualConcept) (SATISFIES
                                                         domainEntityList::1))
          (mapGround IndividualConcept (SATISFIES mapGround df
                                                         ^VIEWSURFACE))
          (domainCoordSys IndividualConcept (SATISFIES domainCoordSys df
                                                         ^CARTESIANSYSTEM))
          (RETURNS (LST OF IndividualConcept) (SATISFIES VALUE:1 df
                                                         ^2DLINEARTRANSFORM)))
(CLISP: MIXED (RECORD (XCoord YCoord)))

```

(* Edited by F.Zdybel on
13-Nov-80.)

(* Determines the appropriate transform to be used to map items from domainCoordinateSystem into the Map, given that the ViewSurface of the map has already been determined. Does this by looking for the maximum spread in x or y domain coordinates among the items and mapping this onto either the height of width (depending upon whether the max spread was along y or x) of the Ground of the Map, minus some reasonable amount, so that no item is exactly on the Boundary of the ViewSurface. As a side effect, the Continuum of the domainCoordinateSystem gets its Exit Role filled with the new created transform, ditto the Entry Role of the Continuum of the Ground.)

```
(for entity in domainEntityList
  bind temp currentEntityLocation minX_MAX.FLOAT
    minY_MAX.FLOAT
    maxX_MIN.FLOAT
    maxY_MIN.FLOAT
    ll ur groundHeight groundWidth
    [mapMarginFactor_ (CAR (KLFindIData $$CONCEPT '(MapMarginFactor)
      adjustedGroundHeight adjustedGroundWidth xMargin yMargin
      scaleFactor xOrigin yOrigin transform
when Position@entity
do (currentEntityLocation_ (CAR ((EstablishDomainEntityPositionInCS
    entity domainCoordinateSystem)
    ;Location))))))
  (minX_ (MIN temp currentEntityLocation:XCoord
    minX))
  (maxX_ (MAX temp maxX))
  (minY_ (MIN temp currentEntityLocation:YCoord
    minY))
  (maxY_ (MAX temp maxY))
  finally [groundWidth_ (fetch XCoord
    of (ur_ (CAR {mapGround;Boundary;UpperRight}))
    )
    - (fetch XCoord of (ll_ (CAR {mapGround;Boundary;LowerLeft}))
    (groundHeight_ur:YCoord-ll:YCoord)
    (adjustedGroundWidth_ groundWidth-(2*(
      xMargin_mapMarginFactor*groundWidth)))
    (adjustedGroundHeight_ groundHeight-(2*(
      yMargin_mapMarginFactor*groundHeight)))
    (scaleFactor_ (MIN adjustedGroundHeight/(maxY-minY)
      adjustedGroundWidth/(maxX-minX)))
    (xOrigin_ll:XCoord+xMargin-scaleFactor*minX)
    (yOrigin_ll:YCoord+yMargin-scaleFactor*minY)
    (transform_ (Make 2DLINEARTRANSFORM From domainCoordSys To
      (CAR {mapGround;Continuum})
      Origin <xOrigin yOrigin> UnitVector
      <scaleFactor scaleFactor>))
    [irole Exit of (atomval domainCoordSys)
```

```

      (vr (atomval transform)))
    [irole Entry of (CAR {mapGround;Continuum})
      (vr (atomval transform))]
    (RETURN transform)))

```

{15}

```

(MakeMapItemLabel?
[DLAMBDA ((application IndividualConcept (SATISFIES application df
                                         `ITEMPLATE))
          (mapGround IndividualConcept (SATISFIES mapGround df
                                         `VIEWSURFACE))
          (RETURNS (LST OF IndividualConcept) (SATISFIES VALUE:1 df
                                         `TEXT)))
          (* Edited by F.Zdybel on
            13-Nov-80.)

```

(* Looks for a Name Role on the first object meta-indicated by the ConceptGroup Role filler of application.
Creates an instance of TEXT based on what it finds.)

```

(PROG ((context (KLGetNamedContext 'AIPSDOMAINMETA))
       (targetRole Name@PHYSOBJECT)
       (domainEntity (FindTemplateObjects application):1))
      (if (for roleMetaIndicator in {application;RoleGroup}
          thereis (KLIsRoleDescendantP (KLGetMetaDescribedEntity
                                         roleMetaIndicator context)
                                         targetRole))
          and (KLFindRoleValues domainEntity targetRole)
          then (RETURN <(Make TEXT String {domainEntity;Name}:1 Ground
                                         mapGround)
                    >)
          else (StopFill))))

```

{16}

```

(MakeMapItems
[DLAMBDA ((mapApplication (LST OF IndividualConcept) (SATISFIES
                                                         (
HomogenousConceptListP mapApplication `ITEMPLATE)))
          (domainEntityList (LST OF IndividualConcept))
          (mapGround IndividualConcept (SATISFIES (KLIsConceptDescendantP
                                                    mapGround
                                                    `VIEWSURFACE)))
          (RETURNS (LST OF IndividualConcept) (SATISFIES
                                                    (HomogenousConceptListP
              VALUE `MAPITEM))))
          (* Edited by F.Zdybel on

```

6-Aug-80.)

(* Goes through the entity list of the Map and determines what kind of Display to individuate for each, based on the value restrictions of the Item roles (For each item in the entity list determines the Application role filler implied by the Application role of the Map, then tries to find some descendant of one of the VR's on mapItemRestrictions that can fulfill the template for the item, or at least fulfill it to the greatest extent possible.))

```
(if (for domainEntity in domainEntityList bind mapItemGeneric
                                     mapItemGenericIA
                                     mapItemGenericIASubjects
first (mapItemGeneric (KLFindVRsOfRole $$ROLE):1)
    (mapItemGenericIA_ { (KLGetAbstraction mapItemGeneric)
                        ;IntendedApplication})
    (mapItemGenericIASubjects_ (FindApplicationObjects
                                mapItemGenericIA))
when (for subject in mapItemGenericIASubjects
    thereis domainEntity df subject)
    and (SupplyConceptMetaIndicator domainEntity
        'AIPSDOMAINMETA)
    ~memb {$$CONCEPT;Item;Application;ConceptGroup}
collect (APPLY* 'Make (KLGetConceptName mapItemGeneric)
            'Application
            (MakeMapItemApplication domainEntity
                                     mapItemGenericIA
                                     mapApplication)
            'Ground mapGround))
else (StopFill))))
```

{17}

```
(MakeMapLabel
[DLAMBDA ((mapName STRINGP)
    (mapGround IndividualConcept (SATISFIES (mapGround df
                                                `VIEWSURFACE)))
    (RETURNS (LST OF IndividualConcept) (SATISFIES (VALUE:1 df
                                                        `TEXT))))
(* Edited by F.Zdybel on
13-Nov-80.)
```

(* Creates an individual of TEXT. At some other subCs of MAP, the map label might be a shaded region with the instance of TEXT superimposed on it, but for the moment we will exercise the ToLocate procedures a bit by putting the label within the Map proper.)

```
<(Make TEXT String mapName Ground mapGround)
>))
```

{18}

```
(MakeMapTransformForMapWithBackground
[DLAMBDA ((transformVR GenericConcept (SATISFIES (transformVR df `MAPPING)
))
(domainCoordSys IndividualConcept (SATISFIES
(domainCoordSys df
`COORDINATESYSTEM)))
(mapGround IndividualConcept (SATISFIES (mapGround df
`VIEWSURFACE)))
(RETURNS (LST OF IndividualConcept) (SATISFIES (VALUE:1 df
MAPPING))))
(* Edited by F.Zdybel on
13-Nov-80.)
```

(* Creates an individual of transformVR, a Generic MAPPING with Origin and UnitVector Roles already restricted to particular LISTP. Since these Value Descriptions will already be inherited by the new individual, there should be no need to copy them onto IRoles. As a side effect, fills the Exit Role of the DomainCoordSys and the Entry Role of the Continuum of the Ground with the new Transform.)

```
(PROG (mapTransform groundCoordSys)
(mapTransform_ (APPLY* 'Make (KLGetConceptName transformVR)
'From domainCoordSys 'To groundCoordSys_
{mapGround;Continuum}:1))
[irole Exit of (atomval domainCoordSys)
(vr (atomval mapTransform))]
[irole Entry of (atomval groundCoordSys)
(vr (atomval mapTransform))]
(RETURN <mapTransform>)))
```

{19}

```
(MakeRectangularMapBorder
DLAMBDA ((mapGroundLowerLeft (LST OF NUMBERP))
(mapGroundUpperRight (LST OF NUMBERP))
(mapGround IndividualConcept (SATISFIES mapGround df
`VIEWSURFACE))
(RETURNS (LST OF IndividualConcept) (SATISFIES VALUE:1 df
`RECTANGLE)))
(* Edited by F.Zdybel on
13-Nov-80.)
```

(* Useful in drawing the border around a flat rectangular Map.
Does this by limning the edge of the ViewSurface that the map
is onto with an instance of RECTANGLE)

```
<(Make RECTANGLE LowerLeft mapGroundLowerLeft UpperRight
  mapGroundUpperRight Ground mapGround)
>])
```

{20}

```
(MakeSimpleMapLegend
  [DLAMBDA ((mapName STRINGP)
    (mapItemList (LST OF IndividualConcept))
    (RETURNS (LST OF IndividualConcept) (SATISFIES (VALUE:1 df
      `MAPLEGEND))))
    (* Edited by F.Zdybel on
      13-Nov-80.)
```

(* Makes the simplest possible kind of Legend
(i.e.: merely a correspondence table between the generic
display forms and the classes of domain world entities they
denote.) Does this by driving the created MapLegend with an
application that designates the ConceptGroup roles of each of
all of the IntendedApplications found for each of the Generic
Items used by the Map.)

```
<(Make MAPLEGEND Label (Make TEXT String (CONCAT "LEGEND for " mapName))
  Application
  (MakeLegendApplication (KLFindVRsOfRole (
KLFindOneNamedGenericRole $$CONCEPT 'Item)):1 mapItemList))
>])
)
```

```
(RPAQQ AIPSMAPUTILITYFNS (FindMapItemPrototypes FindMappingBetweenCSSs
  EstablishDomainEntityPositionInCS
  MakeMapItemApplication
  MapPositionDescription))
(DEFINEQ
```

{21}

```
(FindMapItemPrototypes
  [DLAMBDA ((map IndividualConcept (SATISFIES (map df `MAP))))
    (* Edited by F.Zdybel on
      17-Jul-80.)
```

(* Finds all the possible map item prototypes which apply under the value restrictions of mapPrototype's Item Role.)

```
(for topLevelMapItemRestriction in (KLFindVRsOfRole (
KLFindOneNamedGenericRole map 'Item)) join
  <topLevelMapItemRestriction
    !(KLFindSubConcepts
      topLevelMapItemRestriction)
    >))
```

{22}

```
(FindMappingBetweenCSSs
[DLAMBDA ((coordSys1 IndividualConcept (SATISFIES (coordSys1 df
  ^COORDINATESYSTEM)))
  (coordSys2 IndividualConcept (SATISFIES (coordSys2 df
  ^COORDINATESYSTEM)))
  (RETURNS IndividualConcept (SATISFIES (VALUE df ^MAPPING))))
  (* Edited by F.Zdybel on
  2-Jul-80.)
```

(* Tries to find an existing transformation out of coordSys1 into coordSys2. Should be able to chain together and create subsuming transforms in doing this, but for the moment simply looks for an already existing transform.)

```
(for mapping in {coordSys1;Exit} thereis mapping
  memb {coordSys2;Entry}))
```

{23}

```
(EstablishDomainEntityPositionInCS
[DLAMBDA ((domainEntity IndividualConcept)
  (coordinateSystem IndividualConcept (SATISFIES
    (coordinateSystem df
      ^COORDINATESYSTEM)))
  (RETURNS IndividualConcept (SATISFIES (VALUE df ^POSITION))))
  (* Edited by F.Zdybel on
  9-Jul-80.)
```

(* Insures that one of the Values of domainEntity's Position Role is a position in the given coordinateSystem. If necessary, establishes a new Position IRole for domainEntity as a side-effect.)

```
(OR (for positionDescription in {domainEntity;Position}
```



```

    thereis {positionDescription;Continuum}:1 = coordinateSystem)
  (for positionDescription in {domainEntity;Position}
    bind transform mappedPosition thereis transform (
      FindMappingBetweenCSs
        {positionDescription;Continuum}:1
        coordinateSystem)
    finally mappedPosition_ (MapPositionDescription
      positionDescription
      transform)
    [irole Position of (atomval domainEntity)
      (vr (atomval mappedPosition))]
    (RETURN mappedPosition)))]

```

{24}

```

(MakeMapItemApplication
  [DLAMBDA ((domainEntity IndividualConcept)
    (genericItemIntendedApplication (LST OF IndividualConcept)
      (SATISFIES (HomogenousConceptListP
        genericItemIntendedApplication `TEMPLATE)))
    (mapApplication (LST OF IndividualConcept)
      (SATISFIES HomogenousConceptListP mapApplication
        `TEMPLATE)))
    (RETURNS IndividualConcept (SATISFIES (KLIsConceptDescendantP
      VALUE `ITEMPLATE))))
    (* Edited by F.Zdybel on
      5-Aug-80.)

```

(* Constructs a single ITemplate which describes domainEntity and includes all of roles mentioned in mapApplication in connection with domainEntity that are covered by roles mentioned in genericItemTemplateList in connection with superCs of domainEntity.)

```

(Make ITEMPLATE ConceptGroup (SupplyConceptMetaIndicator domainEntity
  'AIPSDOMAINMETA)
  RoleGroup
  (for role in (FindApplicableApplicationSlots domainEntity
    mapApplication)
    bind (itemApplicationSlots _ (FindApplicableApplicationSlots
      domainEntity
      genericItemIntendedApplication))
    when (for superRole in itemApplicationSlots
      thereis (KLIsRoleDescendantP role superRole))
    collect (SupplyRoleMetaIndicator role 'AIPSDOMAINMETA))))

```

{25}

```

(MapPositionDescription
[DLAMBDA ((position IndividualConcept (SATISFIES (KLIsConceptDescendantP
                                                position `POSITION)))
          (transform IndividualConcept (SATISFIES (KLIsConceptDescendantP
                                                transform `MAPPING)))
          (RETURNS IndividualConcept (SATISFIES (KLIsConceptDescendantP
                                                VALUE `POSITION))))
(CLISP: MIXED (RECORD (XCoord YCoord)))

```

(* Edited by F.Zdybel on
31-Jul-80.)

(* Transforms a location expressed in one coordinate system
via the given transform into a location expressed in a
different coordinate system. The actual transformation is
carried out by the most specific ToTransform procedure
associated with the mapping.)

```

(Make POSITION Location (APPLY* (CAR (KLFindIData transform '(
                                                ToTransform)))
                                transform
                                (CAR {position;Location})))
Continuum
(CAR {transform;To})))
)

```

```

(ADDTOTVAR CKLONEFILES AIPSMAP)
STOP

```

9. SOURCE FILE: AIPSNAVAL

InitENEMYPLANE.....1
InitENEMYSHIP.....2
InitENEMYSUB.....3
InitFRIENDLYPLANE.....4
InitFRIENDLYSHIP.....5
InitFRIENDLYSUB.....6
InitNavalDomainModel.....7
InitPLANE.....8
InitPLATFORM.....9
InitSHIP.....10
InitSUB.....11
InitUNKNOWNPLANE.....12
InitUNKNOWNSHIP.....13
InitUNKNOWNSUB.....14

(FILECREATED "17-Jul-80 02:14:13" <NEWAIPS>AIPSSNAVAL..3 4370

changes to: InitENEMYSHIP InitENEMYSUB

previous date: "13-Jul-80 17:21:25" <NEWAIPS>AIPSSNAVAL..2)

(PRETTYCOMPRINT AIPSSNAVALCOMS)

(RPAQQ AIPSSNAVALCOMS ((FNS * AIPSSNAVALINITFNS)
(ADDVARS (CKLONEFILES AIPSSNAVAL))))

(RPAQQ AIPSSNAVALINITFNS (InitENEMYPLANE InitENEMYSHIP InitENEMYSUB
InitFRIENDLYPLANE InitFRIENDLYSHIP
InitFRIENDLYSUB
InitNavalDomainModel InitPLANE
InitPLATFORM InitSHIP InitSUB
InitUNKNOWNPLANE InitUNKNOWNSHIP
InitUNKNOWNSUB))

(DEFINEQ

{1}

(InitENEMYPLANE
[LAMBDA NIL
[concept ENEMYPLANE
(specializes PLANE PLATFORM)
[roleset NIL
(mods Ownership@PLATFORM)
(vr 'Enemy)]]])

{2}

(InitENEMYSHIP
[LAMBDA NIL
[concept ENEMYSHIP
(specializes SHIP PLATFORM)
[roleset NIL
(mods Ownership@PLATFORM)
(vr 'Enemy)]]])

(* Edited by F.Zdybel on
14-Jul-80.)

{3}

(InitENEMYSUB
[LAMBDA NIL

(* Edited by F.Zdybel on

14-Jul-80.)

```
[concept ENEMYSUB
  (specializes SUB PLATFORM)
  [roleset NIL
    (mods Ownership@PLATFORM)
    (vr 'Enemy)]]])
```

{4}

```
(InitFRIENDLYPLANE
  [LAMBDA NIL
    [concept FRIENDLYPLANE
      (specializes PLANE PLATFORM)
      [roleset NIL
        (mods Ownership@PLATFORM)
        (vr 'Friendly)]]])
```

{5}

```
(InitFRIENDLYSHIP
  [LAMBDA NIL
    [concept FRIENDLYSHIP
      (specializes SHIP PLATFORM)
      [roleset NIL
        (mods Ownership@PLATFORM)
        (vr 'Friendly)]]])
```

{6}

```
(InitFRIENDLYSUB
  [LAMBDA NIL
    [concept FRIENDLYSUB
      (specializes SUB PLATFORM)
      [roleset NIL
        (mods Ownership@PLATFORM)
        (vr 'Friendly)]]])
```

(* Edited by F.Zdybel on
9-Jul-80.)

{7}

```
(InitNavalDomainModel
  [LAMBDA NIL
    (InitPLATFORM)
    (InitSHIP)
    (InitSUB)
    (InitPLANE)
```

(* Knowledge about the domain
of naval tactical platforms.)

```
(InitFRIENDLYSHIP)
(InitFRIENDLYSUB)
(InitFRIENDLYPLANE)
(InitENEMYSHIP)
(InitENEMYSUB)
(InitENEMYPLANE)
(InitUNKNOWNSHIP)
(InitUNKNOWN SUB)
(InitUNKNOWNPLANE])
```

{8}

```
(InitPLANE
[LAMBDA NIL
```

```
(* Edited by F.Zdybel on
11-Jul-80.)
```

```
[concept PLANE
  (specializes VEHICLE)
  [roleset NIL
    (mods Medium@VEHICLE)
    (vr 'Air)]]])
```

{9}

```
(InitPLATFORM
[LAMBDA NIL
```

```
(* Edited by F.Zdybel on
9-Jul-80.)
```

```
[concept PLATFORM
```

```
(* Describes a vehicle with weapons systems and sensor
systems, and an engagement status)
```

```
(specializes VEHICLE)
[roleset EngagementStatus
  (vr ATOM)]
```

```
(* The filler of this role should be either T or NIL depending
on whether the Platform currently has targets engaged or is
itself engaged as a target.)
```

```
[roleset Ownership
  (vr ATOM)]
```

```
(* Expected to be either
Friendly, Enemy or Unknown.))])
```

{10}

```
(InitSHIP
  [LAMBDA NIL
    [concept SHIP
      (specializes VEHICLE)
      [roleset NIL
        (mods Medium@VEHICLE)
        (vr 'Surface)]]])
```

(* Edited by F.Zdybel on
9-Jul-80.)

(* Does not include
submersibles.)

{11}

```
(InitSUB
  [LAMBDA NIL
    [concept SUB
      (specializes VEHICLE)
      [roleset NIL
        (mods Medium@VEHICLE)
        (vr 'SubSurface)]]])
```

{12}

```
(InitUNKNOWNPLANE
  [LAMBDA NIL
    [concept UNKNOWNPLANE
      (specializes PLANE PLATFORM)
      [roleset NIL
        (mods Ownership@PLATFORM)
        (vr 'Unknown)]]])
```

(* Edited by F.Zdybel on
9-Jul-80.)

{13}

```
(InitUNKNOWNSHIP
  [LAMBDA NIL
    [concept UNKNOWNSHIP
      (specializes SHIP PLATFORM)
      [roleset NIL
        (mods Ownership@PLATFORM)
        (vr 'Unknown)]]])
```

{14}

```
(InitUNKNOWNSUB
```

```
[LAMBDA NIL
  [concept UNKNOWNSUB
    (specializes SUB PLATFORM)
    [roleset NIL
      (mods Ownership@PLATFORM)
      (vr 'Unknown)]]])
)

(ADDTOTVAR CKLONEFILES AIPSNAVAL)
STOP
```


10. SOURCE FILE: AIPSNTDS

InitENEMYNTDSPLANE.....1
InitENEMYNTDSSHIP.....2
InitENEMYNTDSSUB.....3
InitFRIENDLYNTDSPLANE.....4
InitFRIENDLYNTDSSHIP.....5
InitFRIENDLYNTDSSUB.....6
InitNTDSConcepts.....7
InitNTDSENGAGEMENTMARK.....8
InitNTDSITEM.....9
InitNTDSMAP.....10
InitNTDSVELOCITYLEADER.....11
InitUNKNOWNNTDSPLANE.....12
InitUNKNOWNNTDSSHIP.....13
InitUNKNOWNNTDSSUB.....14
MakeNTDSEngagementMark?.....15
MakeNTDSGroupStrengthMark?.....16
MakeNTDSVelocityLeader?.....17

(FILECREATED "13-Nov-80 02:21:26" <NEWAIPS>AIPSNTDS..26 31797

changes to: MakeNTDSEngagementMark? MakeNTDSGroupStrengthMark?
MakeNTDSVelocityLeader?

previous date: " 7-Aug-80 19:53:29" <NEWAIPS>AIPSNTDS..25)

(PRETTYCOMPRINT AIPSNTDSCOMS)

(RPAQQ AIPSNTDSCOMS ((FNS * AIPSNTDSINITFNS)
 (FNS * AIPSNTDSDERIVATIONFNS)
 (ADDVARS (CKLONEFILES AIPSNTDS))))

(RPAQQ AIPSNTDSINITFNS (InitENEMYNTDSPLANE InitENEMYNTDSSHIP
 InitENEMYNTDSSUB
 InitFRIENDLYNTDSPLANE
 InitFRIENDLYNTDSSHIP
 InitFRIENDLYNTDSSUB
 InitNTDSConcepts
 InitNTDSENGAGEMENTMARK
 InitNTDSITEM InitNTDSMAP
 InitNTDSVELOCITYLEADER
 InitUNKNOWNNTDSPLANE
 InitUNKNOWNNTDSSHIP
 InitUNKNOWNNTDSSUB))

(DEFINEQ

{1}

(InitENEMYNTDSPLANE
[LAMBDA NIL

(* Edited by F.Zdybel on
13-Jul-80.)

[concept ENEMYNTDSPLANE

(* The symbol used in an
NTDS-type Map to denote an
enemy airplane.)

 (specializes NTDSITEM)
 (abstraction [iconcept ENEMYNTDSPLANEABSTR of DISPLAYABSTRACTION]))
[iconcept ENEMYNTDSPLANEABSTR
 [irole IntendedApplication
 (vr [iconcept ENEMYNTDSPLANEIAPL1]))]
 [irole FixedComponent
 (vr (Make EDGESET VertexList '((-6 0)
 (0 6)
 (6 0))]
 [irole FixedComponent
 (vr (Make POINT Location '(0 0)))]])

```
[iconcept ENEMYNTDSPLANEIAPL1
```

```
(* Indicates that ENEMYNTDSPLANE should apply only to the
depictions of EnemyPlanes. Otherwise, it is very like the
IntendedApplication template for NTDSITEM.)
```

```
(individuates TEMPLATE)
[irole ConceptGroup
  (vr (SupplyConceptMetaIndicator [concept ENEMYPLANE]
    'AIPSDOMAINMETA)))]
[irole RoleGroup
  (vr (SupplyRoleMetaIndicator [roleset Ownership of PLATFORM]
    'AIPSDOMAINMETA)))]
[irole RoleGroup
  (vr (SupplyRoleMetaIndicator [roleset Medium of VEHICLE]
    'AIPSDOMAINMETA)))]
[irole RoleGroup
  (vr (SupplyRoleMetaIndicator [roleset Position of PHYSOBJECT]
    'AIPSDOMAINMETA)))]
[irole RoleGroup
  (vr (SupplyRoleMetaIndicator [roleset EngagementStatus of PLATFORM]
    'AIPSDOMAINMETA)))]
[irole RoleGroup
  (vr (SupplyRoleMetaIndicator [roleset Course of VEHICLE]
    'AIPSDOMAINMETA)))]
[irole RoleGroup
  (vr (SupplyRoleMetaIndicator [roleset Speed of VEHICLE]
    'AIPSDOMAINMETA)))]
[irole RoleGroup
  (vr (SupplyRoleMetaIndicator [roleset Name of PHYSOBJECT]
    'AIPSDOMAINMETA))))]
```

```
{2}
```

```
(InitENEMYNTDSSHIP
[LAMBDA NIL
```

```
(* Edited by F.Zdybel on
13-Jul-80.)
```

```
[concept ENEMYNTDSSHIP
```

```
(* The symbol used in an NTDS-type Map to denote a enemy
surface ship.)
```

```
(specializes NTDSITEM)
(abstraction [iconcept ENEMYNTDSSHIPABSTR of DISPLAYABSTRACTION]))
[iconcept ENEMYNTDSSHIPABSTR
[irole IntendedApplication
```

```

(vr [iconcept ENEMYNTDSSHIAPL1]))
[irole FixedComponent
  (vr (Make EDGESET VertexList '((-6 0)
    (0 6)
    (6 0)
    (0 -6)
    (-6 0))
    Location '(0 0)))]
[irole FixedComponent
  (vr (Make POINT Location '(0 0)))]
[iconcept ENEMYNTDSSHIAPL1

```

(* Indicates that ENEMYNTDSSHIP applies only to the depiction of EnemyShips. Otherwise it is much like the IntendedApplication template for NTDSITEM.)

```

(individuates TEMPLATE)
[irole ConceptGroup
  (vr (SupplyConceptMetaIndicator [concept ENEMYSHIP]
    'AIPSDOMAINMETA)))]
[irole RoleGroup
  (vr (SupplyRoleMetaIndicator [roleset Ownership of PLATFORM]
    'AIPSDOMAINMETA)))]
[irole RoleGroup
  (vr (SupplyRoleMetaIndicator [roleset Medium of VEHICLE]
    'AIPSDOMAINMETA)))]
[irole RoleGroup
  (vr (SupplyRoleMetaIndicator [roleset Position of PHYSOBJECT]
    'AIPSDOMAINMETA)))]
[irole RoleGroup
  (vr (SupplyRoleMetaIndicator [roleset EngagementStatus of PLATFORM]
    'AIPSDOMAINMETA)))]
[irole RoleGroup
  (vr (SupplyRoleMetaIndicator [roleset Course of VEHICLE]
    'AIPSDOMAINMETA)))]
[irole RoleGroup
  (vr (SupplyRoleMetaIndicator [roleset Speed of VEHICLE]
    'AIPSDOMAINMETA)))]
[irole RoleGroup
  (vr (SupplyRoleMetaIndicator [roleset Name of PHYSOBJECT]
    'AIPSDOMAINMETA)))]])

```

{3}

```

(InitENEMYNTDSSUB
[LAMBDA NIL

```

```

[concept ENEMYNTDSSUB

```

(* Edited by F.Zdybel on
13-Jul-80.)

(* The symbol used in an NTDS-type Map to denote an enemy submarine.)

```
(specializes NTDSITEM)
(abstraction [iconcept ENEMYNTDSSUBABSTR of DISPLAYABSTRACTION]))
[iconcept ENEMYNTDSSUBABSTR
  [irole IntendedApplication
    (vr [iconcept ENEMYNTDSSUBIAPL1])]
  [irole FixedComponent
    (vr (Make EDGESET VertexList '((-6 0)
      (-6 -6)
      (6 0))
    [irole FixedComponent
      (vr (Make POINT Location '(0 0)))]
    [iconcept ENEMYNTDSSUBIAPL1
```

(* Indicates that ENEMYNTDSSUB should apply only to the depictions of EnemySubs. Otherwise, it is very like the IntendedApplication template for NTDSITEM.)

```
(individuates TEMPLATE)
[irole ConceptGroup
  (vr (SupplyConceptMetaIndicator [concept ENEMYSUB]
    'AIPSDOMAINMETA))]
[irole RoleGroup
  (vr (SupplyRoleMetaIndicator [roleset Ownership of PLATFORM]
    'AIPSDOMAINMETA))]
[irole RoleGroup
  (vr (SupplyRoleMetaIndicator [roleset Medium of VEHICLE]
    'AIPSDOMAINMETA))]
[irole RoleGroup
  (vr (SupplyRoleMetaIndicator [roleset Position of PHYSOBJECT]
    'AIPSDOMAINMETA))]
[irole RoleGroup
  (vr (SupplyRoleMetaIndicator [roleset EngagementStatus of PLATFORM]
    'AIPSDOMAINMETA))]
[irole RoleGroup
  (vr (SupplyRoleMetaIndicator [roleset Course of VEHICLE]
    'AIPSDOMAINMETA))]
[irole RoleGroup
  (vr (SupplyRoleMetaIndicator [roleset Speed of VEHICLE]
    'AIPSDOMAINMETA))]
[irole RoleGroup
  (vr (SupplyRoleMetaIndicator [roleset Name of PHYSOBJECT]
    'AIPSDOMAINMETA)))]
```

{4}

```
(InitFRIENDLYNTDSPLANE
[LAMBDA NIL
```

```
(* Edited by F.Zdybel on
13-Jul-80.)
```

```
[concept FRIENDLYNTDSPLANE
```

```
(* The symbol used in an
NTDS-type Map to denote a
friendly airplane.)
```

```
(specializes NTDSITEM)
(abstraction [iconcept FRIENDLYNTDSPLANEABSTR of DISPLAYABSTRACTION]))
[iconcept FRIENDLYNTDSPLANEABSTR
[irole IntendedApplication
(vr [iconcept FRIENDLYNTDSPLANEIAPL1])]
[irole FixedComponent
(vr (Make EDGASET VertexList '((-6.0 0.0)
(-5.706339 1.854102)
(-4.854102 3.526712)
(-3.526712 4.854102)
(-1.854102 5.706339)
(0.0 6.0)
(1.854102 5.706339)
(3.526712 4.854102)
(4.854102 3.526712)
(5.706339 1.854102)
(6.0 0.0))]
[irole FixedComponent
(vr (Make POINT Location '(0 0)))]
[iconcept FRIENDLYNTDSPLANEIAPL1
```

```
(* Indicates that FRIENDLYNTDSPLANE should apply only to the
depictions of FriendlyPlanes. Otherwise, it is very like the
IntendedApplication template for NTDSITEM.)
```

```
(individuates TEMPLATE)
[irole ConceptGroup
(vr (SupplyConceptMetaIndicator [concept FRIENDLYPLANE]
'AIPSDOMAINMETA))]
[irole RoleGroup
(vr (SupplyRoleMetaIndicator [roleset Ownership of PLATFORM]
'AIPSDOMAINMETA))]
[irole RoleGroup
(vr (SupplyRoleMetaIndicator [roleset Medium of VEHICLE]
'AIPSDOMAINMETA))]
[irole RoleGroup
(vr (SupplyRoleMetaIndicator [roleset Position of PHYSOBJECT]
'AIPSDOMAINMETA))]
[irole RoleGroup
(vr (SupplyRoleMetaIndicator [roleset EngagementStatus of PLATFORM]
'AIPSDOMAINMETA))]
[irole RoleGroup
```

```

      (vr (SupplyRoleMetaIndicator [roleset Course of VEHICLE]
                                     'AIPSDOMAINMETA)))]
[irole RoleGroup
  (vr (SupplyRoleMetaIndicator [roleset Speed of VEHICLE]
                               'AIPSDOMAINMETA)))]
[irole RoleGroup
  (vr (SupplyRoleMetaIndicator [roleset Name of PHYSOJECT]
                               'AIPSDOMAINMETA))]])]

```

{5}

```

(InitFRIENDLYNTDSSHIP
[LAMBDA NIL

```

```

(* Edited by F.Zdybel on
13-Jul-80.)

```

```

[concept FRIENDLYNTDSSHIP

```

```

(* The symbol used in an NTDS-type Map to denote a friendly
surface ship.)

```

```

      (specializes NTDSITEM)
      (abstraction [iconcept FRIENDLYNTDSSHIPABSTR of DISPLAYABSTRACTION]))]
[iconcept FRIENDLYNTDSSHIPABSTR
  [irole IntendedApplication
    (vr [iconcept FRIENDLYNTDSSHIPIAPL1]))]
  [irole FixedComponent
    (vr (Make CIRCLE Radius 6 Location '(0 0)))]
  [irole FixedComponent
    (vr (Make POINT Location '(0 0)))]
[iconcept FRIENDLYNTDSSHIPIAPL1

```

```

(* Indicates that FRIENDLYNTDSSHIP should apply only to the
depictions of FriendlyShips. Otherwise, it is very like the
IntendedApplication template for NTDSITEM.)

```

```

(individuates TEMPLATE)
[irole ConceptGroup
  (vr (SupplyConceptMetaIndicator [concept FRIENDLYSHIP]
                                   'AIPSDOMAINMETA)))]
[irole RoleGroup
  (vr (SupplyRoleMetaIndicator [roleset Ownership of PLATFORM]
                               'AIPSDOMAINMETA)))]
[irole RoleGroup
  (vr (SupplyRoleMetaIndicator [roleset Medium of VEHICLE]
                               'AIPSDOMAINMETA)))]
[irole RoleGroup
  (vr (SupplyRoleMetaIndicator [roleset Position of PHYSOJECT]

```

```

                                'AIPSDOMAINMETA)))
[irole RoleGroup
  (vr (SupplyRoleMetaIndicator [roleset EngagementStatus of PLATFORM]
                                'AIPSDOMAINMETA)))
[irole RoleGroup
  (vr (SupplyRoleMetaIndicator [roleset Course of VEHICLE]
                                'AIPSDOMAINMETA)))
[irole RoleGroup
  (vr (SupplyRoleMetaIndicator [roleset Speed of VEHICLE]
                                'AIPSDOMAINMETA)))
[irole RoleGroup
  (vr (SupplyRoleMetaIndicator [roleset Name of PHYSOJECT]
                                'AIPSDOMAINMETA))))])

```

{6}

```

(InitFRIENDLYNTDSSUB
[LAMBDA NIL

```

```

(* Edited by F.Zdybel on
13-Jul-80.)

```

```

[concept FRIENDLYNTDSSUB

```

```

(* The symbol used in an
NTDS-type Map to denote a
friendly submarine.)

```

```

  (specializes NTDSITEM)
  (abstraction [iconcept FRIENDLYNTDSSUBABSTR of DISPLAYABSTRACTION]))
[iconcept FRIENDLYNTDSSUBABSTR
  [irole IntendedApplication
    (vr [iconcept FRIENDLYNTDSSUBIAPL1]))]
  [irole FixedComponent
    (vr (Make EDGESET VertexList '((-6.0 0.0)
      (-5.706339 -1.854102)
      (-4.854102 -3.526712)
      (-3.526712 -4.854102)
      (-1.854102 -5.706339)
      (0.0 -6.0)
      (1.854102 -5.706339)
      (3.526711 -4.854102)
      (4.854102 -3.526711)
      (5.706339 -1.854102)
      (6.0 0.0))
    [irole FixedComponent
      (vr (Make POINT Location '(0 0)))]])
[iconcept FRIENDLYNTDSSUBIAPL1

```

```

(* Indicates that FRIENDLYNTDSSUB should apply only to the
depictions of FriendlySubs. Otherwise, it is very like the
IntendedApplication template for NTDSITEM.)

```



```

(individuates TEMPLATE)
[irole ConceptGroup
  (vr (SupplyConceptMetaIndicator [concept FRIENDLYSUB]
    'AIPSDOMAINMETA)))]
[irole RoleGroup
  (vr (SupplyRoleMetaIndicator [roleset Ownership of PLATFORM]
    'AIPSDOMAINMETA)))]
[irole RoleGroup
  (vr (SupplyRoleMetaIndicator [roleset Medium of VEHICLE]
    'AIPSDOMAINMETA)))]
[irole RoleGroup
  (vr (SupplyRoleMetaIndicator [roleset Position of PHYSOBJECT]
    'AIPSDOMAINMETA)))]
[irole RoleGroup
  (vr (SupplyRoleMetaIndicator [roleset EngagementStatus of PLATFORM]
    'AIPSDOMAINMETA)))]
[irole RoleGroup
  (vr (SupplyRoleMetaIndicator [roleset Course of VEHICLE]
    'AIPSDOMAINMETA)))]
[irole RoleGroup
  (vr (SupplyRoleMetaIndicator [roleset Speed of VEHICLE]
    'AIPSDOMAINMETA)))]
[irole RoleGroup
  (vr (SupplyRoleMetaIndicator [roleset Name of PHYSOBJECT]
    'AIPSDOMAINMETA))]]]]

```

{7}

```

(InitNTDSConcepts
[LAMBDA NIL

```

```

(* Edited by F.Zdybel on
14-Jul-80.)
(* Initializes the displays
used for NTDS-type tactical
maps.)

```

```

(InitNTDSMAP)
(InitNTDSITEM)
(InitFRIENDLYNTDSSHIP)
(InitFRIENDLYNTDSPLANE)
(InitFRIENDLYNTDSSUB)
(InitENEMYNTDSSHIP)
(InitENEMYNTDSSUB)
(InitENEMYNTDSPLANE)
(InitUNKNOWNNTDSSHIP)
(InitUNKNOWNNTDSSUB)
(InitUNKNOWNNTDSPLANE)
(InitNTDSENGAGEMENTMARK)
(InitNTDSVELOCITYLEADER))

```

{8}

```
(InitNTDSENGAGEMENTMARK
[LAMBDA NIL
```

```
(* Edited by F.Zdybel on
17-Jul-80.)
```

```
[concept NTDSENGAGEMENTMARK
```

```
(* This is the little horizontal line through the locator dot
in an NTDS symbol which indicates that the depicted platform
has engaged or been engaged by some other platform.)
```

```
(specializes DISPLAY)
(abstraction [iconcept NTDSENGAGEMENTMARKABSTR of DISPLAYABSTRACTION])
]
[iconcept NTDSENGAGEMENTMARKABSTR
[irole FixedComponent
[vr (Make LINESEGMENT EndPoint '((-6 0)
(6 0))]
[irole IntendedApplication
(vr [iconcept NTDSENGAGEMENTMARKIAPL1])]]]
[iconcept NTDSENGAGEMENTMARKIAPL1
(individuates TEMPLATE)
[irole ConceptGroup
(vr (SupplyConceptMetaIndicator [concept PLATFORM]
'AIPSDOMAINMETA))]
[irole RoleGroup
(vr (SupplyRoleMetaIndicator [roleset EngagementStatus of PLATFORM]
'AIPSDOMAINMETA))]
[concept NARROWNTDSENGAGEMENTMARK
```

```
(* Because the box on which the NTDS Symbols for unknown
platforms are based is one unit narrower than the circle for
friendlys or the lozenge for enemies, it is necessary to have
a separate type of engagement mark for the unknowns.)
```

```
(specializes NTDSENGAGEMENTMARK)
(abstraction [iconcept NARROWNTDSENGAGEMENTMARKABSTR of
DISPLAYABSTRACTION])
[iconcept NARROWNTDSENGAGEMENTMARKABSTR
[irole FixedComponent
[vr (Make LINESEGMENT EndPoint '((-5 0)
(5 0))]
[irole IntendedApplication
(vr [iconcept NTDSENGAGEMENTMARKIAPL1])]]]]
```

```
{9}
```

(InitNTDSITEM
[LAMBDA NIL

(* Edited by F.Zdybel on
7-Aug-80.)

[concept NTDSITEM

(* A Map Item that is one of
the NTDS symbols.)

(specializes MAPITEM)
(abstraction [iconcept NTDSITEMABSTR of DISPLAYABSTRACTION])

[roleset NIL

(mods Application@DISPLAY)

(derivation (Consequents EngagementMark VelocityLeader
GroupStrengthMarker)))

[roleset VelocityLeader

(diffs Realization@DISPLAY)

(vr NTDSVELOCITYLEADER)

(derivation (MakeNTDSVelocityLeader? (Prerequisites Application
Ground)

(Arguments

{ \$\$CONCEPT;Application}:1

{ \$\$CONCEPT;Ground}:1)))

(* This is the vector line that extends from the center of the
NTDSItem, the length and orientation of which denote the
velocity of the vehicle.)

[roleset EngagementMark

(diffs Realization@DISPLAY)

(vr NTDSENGAGEMENTMARK)

(derivation (MakeNTDSEngagementMark? (Prerequisites Application
Ground)

(Arguments

{ \$\$CONCEPT;Application}:1

{ \$\$CONCEPT;Ground}:1)))

(* This is the line through the center of the NTDSItem, which
indicates whether the denoted vehicle has engaged
(or been engaged by) as the target of some other vehicle)

[roleset GroupStrengthMarker

(diffs Realization@DISPLAY)

(vr NTDSSTRENGTHMARK)

(derivation (MakeNTDSGroupStrengthMark? (Prerequisites Application)
(Arguments
{ \$\$CONCEPT;Application})))

(* This is the little hashmark through the left side of the

NTDSItem which indicates whether it denotes one, few, or many vehicles.)

]

[iconcept NTDSITEMIAPL1

(* This template is used to indicate the intended application not only of NTDSITEM, but also in many cases the intended application of generic descendants of NTDSITEM (e.g.: FRIENDLYNTDSSHIP, ENEMYNTDSSUB.) The possible purposes of these little Displays include depicting the ownership, location, velocity, name, and engagement status of various vehicles. Of course, these symbols also display the general type of the vehicle (e.g., Ship, Sub), but at the moment there is no RoleSet on VEHICLE to redundantly echo the category, thus there is no way to get a grip on that particular information with a Template. This is probably a deficiency in Template.)

(individuates TEMPLATE)

[irole ConceptGroup

(vr (SupplyConceptMetaIndicator [concept SHIP]
'AIPSDOMAINMETA))]

[irole ConceptGroup

(vr (SupplyConceptMetaIndicator [concept SUB]
'AIPSDOMAINMETA))]

[irole ConceptGroup

(vr (SupplyConceptMetaIndicator [concept PLANE]
'AIPSDOMAINMETA))]

[irole RoleGroup

(vr (SupplyRoleMetaIndicator [roleset Ownership of PLATFORM]
'AIPSDOMAINMETA))]

[irole RoleGroup

(vr (SupplyRoleMetaIndicator [roleset Medium of VEHICLE]
'AIPSDOMAINMETA))]

[irole RoleGroup

(vr (SupplyRoleMetaIndicator [roleset Position of PHYSOBJECT]
'AIPSDOMAINMETA))]

[irole RoleGroup

(vr (SupplyRoleMetaIndicator [roleset EngagementStatus of PLATFORM]
'AIPSDOMAINMETA))]

[irole RoleGroup

(vr (SupplyRoleMetaIndicator [roleset Course of VEHICLE]
'AIPSDOMAINMETA))]

[irole RoleGroup

(vr (SupplyRoleMetaIndicator [roleset Speed of VEHICLE]
'AIPSDOMAINMETA))]

[irole RoleGroup

(vr (SupplyRoleMetaIndicator [roleset Name of PHYSOBJECT]

'AIPSDOMAINMETA))]]))

{10}

(InitNTDSMAP
[LAMBDA NIL

[concept NTDSMAP

(* Edited by F.Zdybel on
6-Aug-80.)(* Intended to describe the
familiar type of NTDS situation
display.)

```

(specializes RECTANGULARMAP)
[roleset NIL
  (mods ScaleUnit@RECTANGULARMAP)
  (itags (DefaultFiller `NM))]
[roleset FriendlyShipSym
  (diffs Item@MAP)
  (modality Obligatory)
  (number (0 NIL))
  (vr FRIENDLYNTDSSHIP)]
[roleset FriendlySubSym
  (diffs Item@MAP)
  (modality Obligatory)
  (number (0 NIL))
  (vr FRIENDLYNTDSSUB)]
[roleset FriendlyPlaneSym
  (diffs Item@MAP)
  (modality Obligatory)
  (number (0 NIL))
  (vr FRIENDLYNTDSPLANE)]
[roleset EnemyShipSym
  (diffs Item@MAP)
  (modality Obligatory)
  (number (0 NIL))
  (vr ENEMYNTDSSHIP)]
[roleset EnemySubSym
  (diffs Item@MAP)
  (modality Obligatory)
  (number (0 NIL))
  (vr ENEMYNTDSSUB)]
[roleset EnemyPlaneSym
  (diffs Item@MAP)
  (modality Obligatory)
  (number (0 NIL))
  (vr ENEMYNTDSPLANE)]
[roleset UnknownShipSym
  (diffs Item@MAP)
  (modality Obligatory)
  (number (0 NIL))
  (vr UNKNOWNNTDSSHIP)]

```

```

[roleset UnknownSubSym
  (diffs Item@MAP)
  (modality Obligatory)
  (number (0 NIL))
  (vr UNKNOWNNTDSSUB)]
[roleset UnknownPlaneSym
  (diffs Item@MAP)
  (modality Obligatory)
  (number (0 NIL))
  (vr UNKNOWNNTDSPLANE)]])

```

{11}

```

(InitNTDSVELOCITYLEADER
[LAMBDA NIL

```

(* Edited by F.Zdybel on
6-Aug-80.)

```

[concept NTDSVELOCITYLEADER

```

(* This is the line leading out from the locator dot of an
NTDS Symbol which gives the course and speed of the platform)

```

  (specializes DISPLAY)
  (abstraction [iconcept NTDSVELOCITYLEADERABSTR of DISPLAYABSTRACTION])
  [roleset Leader
    (modality Obligatory)
    (mods Realization@DISPLAY)
    (vr LINESEGMENT)
    (itags (DefaultFiller (Make LINESEGMENT))))]
[iconcept NTDSVELOCITYLEADERABSTR
  [irole IntendedApplication
    (vr [iconcept NTDSVELOCITYLEADERIAPL1])]]
[iconcept NTDSVELOCITYLEADERIAPL1
  (individuates TEMPLATE)
  [irole ConceptGroup
    (vr (SupplyConceptMetaIndicator [concept PLATFORM]
      'AIPSDOMAINMETA)))]
  [irole RoleGroup
    (vr (SupplyRoleMetaIndicator [roleset Course of VEHICLE]
      'AIPSDOMAINMETA)))]
  [irole RoleGroup
    (vr (SupplyRoleMetaIndicator [roleset Speed of VEHICLE]
      'AIPSDOMAINMETA)))]])

```

{12}

```

(InitUNKNOWNNTDSPLANE
[LAMBDA NIL

```

(* Edited by F.Zdybel on

7-Aug-80.)

[concept UNKNOWNNTDSPLANE

(* The symbol used in an
NTDS-type Map to denote an
unknown airplane.)

```

    (specializes NTDSITEM)
    (abstraction [iconcept UNKNOWNNTDSPLANEABSTR of DISPLAYABSTRACTION])
    [roleset NIL
      (mods EngagementMark@NTDSITEM)
      (vr NARROWNTDSENGAGEMENTMARK)]
    [iconcept UNKNOWNNTDSPLANEABSTR
      [irole IntendedApplication
        (vr [iconcept UNKNOWNNTDSPLANEIAPL1])]
      [irole FixedComponent
        (vr (Make EDGESET VertexList '((-5 0)
          (-5 5)
          (5 5)
          (5 0))
        [irole FixedComponent
          (vr (Make POINT Location '(0 0))))]]
    [iconcept UNKNOWNNTDSPLANEIAPL1

```

(* Indicates that UNKNOWNNTDSPLANE should apply only to the depictions of UnknownPlanes. Otherwise, it is very like the IntendedApplication template for NTDSITEM.)

```

(individuates TEMPLATE)
[irole ConceptGroup
  (vr (SupplyConceptMetaIndicator [concept UNKNOWNPLANE]
    'AIPSDOMAINMETA))]
[irole RoleGroup
  (vr (SupplyRoleMetaIndicator [roleset Ownership of PLATFORM]
    'AIPSDOMAINMETA))]
[irole RoleGroup
  (vr (SupplyRoleMetaIndicator [roleset Medium of VEHICLE]
    'AIPSDOMAINMETA))]
[irole RoleGroup
  (vr (SupplyRoleMetaIndicator [roleset Position of PHYSOBJECT]
    'AIPSDOMAINMETA))]
[irole RoleGroup
  (vr (SupplyRoleMetaIndicator [roleset EngagementStatus of PLATFORM]
    'AIPSDOMAINMETA))]
[irole RoleGroup
  (vr (SupplyRoleMetaIndicator [roleset Course of VEHICLE]
    'AIPSDOMAINMETA))]
[irole RoleGroup
  (vr (SupplyRoleMetaIndicator [roleset Speed of VEHICLE]
    'AIPSDOMAINMETA))]
[irole RoleGroup

```

```
(vr (SupplyRoleMetaIndicator [roleset Name of PHYSOJECT]
                              'AIPSDOMAINMETA))]]])
```

{13}

```
(InitUNKNOWNNTDSSHIP
[LAMBDA NIL
```

```
(* Edited by F.Zdybel on
7-Aug-80.)
```

```
[concept UNKNOWNNTDSSHIP
```

```
(* Can be used in an NTDSMAP to depict unclassified Ships, or
Ships that have been classified as Unknown
(as indicated by their Ownership Role.))
```

```
(specializes NTDSITEM)
(abstraction [iconcept UNKNOWNNTDSSHIPABSTR of DISPLAYABSTRACTION])
[roleset NIL
  (mods EngagementMark@NTDSITEM)
  (vr NARROWNTDSENGAGEMENTMARK))]
[iconcept UNKNOWNNTDSSHIPABSTR
  [irole IntendedApplication
    (vr [iconcept UNKNOWNNTDSSHIPIAPL1])]
  [irole FixedComponent
    (vr (Make EDGESET VertexList '((-5 -5)
                                   (-5 5)
                                   (5 5)
                                   (5 -5)
                                   (-5 -5))
        Location '(0 0)))]
  [irole FixedComponent
    (vr (Make POINT Location '(0 0)))]
[iconcept UNKNOWNNTDSSHIPIAPL1
```

```
(* Indicates that UNKNOWNNTDSSHIP should apply only to the
depictions of UnknownShips. Otherwise, it is very like the
IntendedApplication template for NTDSITEM)
```

```
(individuates TEMPLATE)
[irole ConceptGroup
  (vr (SupplyConceptMetaIndicator [concept UNKNOWNSHIP]
                                  'AIPSDOMAINMETA))]
[irole RoleGroup
  (vr (SupplyRoleMetaIndicator [roleset Ownership of PLATFORM]
                              'AIPSDOMAINMETA))]
[irole RoleGroup
  (vr (SupplyRoleMetaIndicator [roleset Medium of VEHICLE]
```



```

'AIPSDOMAINMETA)))
[irole RoleGroup
  (vr (SupplyRoleMetaIndicator [roleset Position of PHYSOBJECT]
    'AIPSDOMAINMETA))]
[irole RoleGroup
  (vr (SupplyRoleMetaIndicator [roleset EngagementStatus of PLATFORM]
    'AIPSDOMAINMETA))]
[irole RoleGroup
  (vr (SupplyRoleMetaIndicator [roleset Course of VEHICLE]
    'AIPSDOMAINMETA))]
[irole RoleGroup
  (vr (SupplyRoleMetaIndicator [roleset Speed of VEHICLE]
    'AIPSDOMAINMETA))]
[irole RoleGroup
  (vr (SupplyRoleMetaIndicator [roleset Name of PHYSOBJECT]
    'AIPSDOMAINMETA))]]))

```

{14}

```

(InitUNKNOWNNTDSSUB
  [LAMBDA NIL

```

```

(* Edited by F.Zdybel on
7-Aug-80.)

```

```

[concept UNKNOWNNTDSSUB

```

```

(* The symbol used in an
NTDS-type Map to denote an
unknown submarine.)

```

```

  (specializes NTDSITEM)
  (abstraction [iconcept UNKNOWNNTDSSUBABSTR of DISPLAYABSTRACTION])
  [roleset NIL
    (mods EngagementMark@NTDSITEM)
    (vr NARROWNTDSENGAGEMENTMARK)]]
[iconcept UNKNOWNNTDSSUBABSTR
  [irole IntendedApplication
    (vr [iconcept UNKNOWNNTDSSUBIAPL1])]
  [irole FixedComponent
    [vr (Make EDGESET VertexList '((-5 0)
      (-5 -5)
      (5 -5)
      (5 0))
    [irole FixedComponent
      (vr (Make POINT Location '(0 0)))]
  [iconcept UNKNOWNNTDSSUBIAPL1

```

```

(* Indicates that UNKNOWNNTDSSUB should apply only to the
depictions of UnknownSubs. Otherwise, it is very like the
IntendedApplication template for NTDSITEM.)

```

```

(individuates TEMPLATE)

```

```

[irole ConceptGroup
  (vr (SupplyConceptMetaIndicator [concept UNKNOWNSUB]
                                     'AIPSDOMAINMETA)))]
[irole RoleGroup
  (vr (SupplyRoleMetaIndicator [roleset Ownership of PLATFORM]
                               'AIPSDOMAINMETA)))]
[irole RoleGroup
  (vr (SupplyRoleMetaIndicator [roleset Medium of VEHICLE]
                               'AIPSDOMAINMETA)))]
[irole RoleGroup
  (vr (SupplyRoleMetaIndicator [roleset Position of PHYSOBJECT]
                               'AIPSDOMAINMETA)))]
[irole RoleGroup
  (vr (SupplyRoleMetaIndicator [roleset EngagementStatus of PLATFORM]
                               'AIPSDOMAINMETA)))]
[irole RoleGroup
  (vr (SupplyRoleMetaIndicator [roleset Course of VEHICLE]
                               'AIPSDOMAINMETA)))]
[irole RoleGroup
  (vr (SupplyRoleMetaIndicator [roleset Speed of VEHICLE]
                               'AIPSDOMAINMETA)))]
[irole RoleGroup
  (vr (SupplyRoleMetaIndicator [roleset Name of PHYSOBJECT]
                               'AIPSDOMAINMETA)))]])
)

```

```

(RPAQQ AIPSNNTDSDERIVATIONFNS (MakeNTDSEngagementMark?
                               MakeNTDSGroupStrengthMark?
                               MakeNTDSVelocityLeader?))
(DEFINEQ

```

{15}

```

(MakeNTDSEngagementMark?
[DLAMBDA ((template IndividualConcept (SATISFIES template df `ITEMPLATE))
          (mapGround IndividualConcept (SATISFIES mapGround df
                                                  `VIEWSURFACE)))
  (RETURNS (LST OF IndividualConcept) (SATISFIES VALUE:1 df
                                                  `NTDSENGAGEMENTMARK)))
  (* Edited by F.Zdybel on
  13-Nov-80.)

```

(* Looks through the templates in applicationList (the Application Role fillers at the NTDSItem) and at the EngagementStatus of the depicted vehicle and determines whether or not to create an engagement mark as part of the NTDSItem. If the role should not be filled, the derivation is aborted. Otherwise, an instance of NTDSENGAGEMENTMARK or NARROWNTDSENGAGEMENTMARK is returned.)

```

(PROG ((vehicle ((FindTemplateObjects template):1))
      (targetRole (EngagementStatus@PLATFORM))
      (context (KLGetNamedContext 'AIPSDOMAINMETA)))
      (if (for roleMetaIndicator in {template;RoleGroup}
          thereis (KLIsRoleDescendantP (KLGetMetaDescribedEntity
                                         roleMetaIndicator context)
                                         targetRole))
          and (KLFindRoleValues vehicle targetRole):1~=NIL
          then (RETURN <(APPLY* 'Make (KLGetConceptName (CAR
                                                         { $$PROTOTYPE;EngagementMark}))
                             'Application
                             (Make ITEMPLATE ConceptGroup
                              (SupplyConceptMetaIndicator
                               vehicle 'AIPSDOMAINMETA)
                              RoleGroup
                              (SupplyRoleMetaIndicator
                               targetRole
                               'AIPSDOMAINMETA))
                             'Ground mapGround)
              >)
          else (StopFill))))])

```

{16}

```

(MakeNTDSGroupStrengthMark?
 [DLAMBDA ((itemApplications (LST OF IndividualConcept) (SATISFIES
                                                         (
HomogenousConceptListP itemApplications `ITEMPLATE)))
          (RETURNS (LST OF IndividualConcept) (SATISFIES VALUE:1 df
                                                         `DISPLAYITEM)))
          (* Edited by F.Zdybel on
          13-Nov-80.)
          (StopDerivation)])

```

{17}

```

(MakeNTDSVelocityLeader?
 [DLAMBDA ((template IndividualConcept (SATISFIES (template df
                                                         `ITEMPLATE)))
          (mapGround IndividualConcept (SATISFIES mapGround df
                                                         `VIEWSURFACE))
          (RETURNS (LST OF IndividualConcept) (SATISFIES VALUE:1 df
                                                         `NTDSVELOCITYLEADER)))
          (* Edited by F.Zdybel on
          13-Nov-80.)

```

(* Looks through the templates in applicationList
 (the Application Role fillers at the NTDSItem) and determines

whether or not to create a velocity leader as part of the NTDSItem. If the templates do not specify the course and the speed of the object being depicted, then the derivation is aborted. Otherwise, an instance of LINESEGMENT is returned.)

```
(PROG ((vehicle (FindTemplateObjects template):1)
      (speedRole Speed@VEHICLE)
      (courseRole Course@VEHICLE)
      (context (KLGetNamedContext 'AIPSDOMAINMETA)))
  (if (for roleMetaIndicator in {template;RoleGroup}
      thereis (KLIsRoleDescendantP (KLGetMetaDescribedEntity
                                     roleMetaIndicator context)
                                     courseRole))
      and (for roleMetaIndicator in {template;RoleGroup}
          thereis (KLIsRoleDescendantP (
KLGetMetaDescribedEntity roleMetaIndicator context)
                                     speedRole))
          and (KLFindRoleValues vehicle courseRole)
          and (KLFindRoleValues vehicle speedRole)
      then (RETURN <(Make NTDSVELOCITYLEADER Application
                    (Make ITEMPLATE ConceptGroup
                     (SupplyConceptMetaIndicator vehicle
                      'AIPSDOMAINMETA)
                     RoleGroup
                     <(SupplyRoleMetaIndicator
                      courseRole
                      'AIPSDOMAINMETA)
                      (SupplyRoleMetaIndicator
                      speedRole
                      'AIPSDOMAINMETA)
                     >)
                    Ground mapGround)
          >)
      else (StopFill))))))
)

(ADDTOTVAR CKLONEFILES AIPSNTDS)
STOP
```

11. SOURCE FILE: AIPSREALIZATION

InitCIRCLE.....	1
InitCLOSEDCURVE.....	2
InitCURVE.....	3
InitDISPLAYATOM.....	4
InitDISPLAYCOMPOSITE.....	5
InitDISPLAYFORM.....	6
InitDisplayFormConcepts.....	7
InitEDGESET.....	8
InitELLIPSE.....	9
InitLINESEGMENT.....	10
InitPOINT.....	11
InitPOLYGON.....	12
InitRealizationModel.....	15
InitRECTANGLE.....	13
InitREGULARPOLYGON.....	14
MakeRectangle.....	18
TLDisplayAtom.....	16
TLRectangle.....	17
TMRectangle.....	19
TSRectangle.....	20

(FILECREATED "29-Jan-81 01:10:48" <NEWAIPS>AIPSREALIZATION..28 11002

changes to: TLDisplayAtom

previous date: "13-Jan-81 22:59:22" <NEWAIPS>AIPSREALIZATION..27)

(PRETTYCOMPRINT AIPSREALIZATIONCOMS)

```
(RPAQQ AIPSREALIZATIONCOMS ((* The AIPS Realization Model resides here.
                             This contains the lowest-level description
                             of drawable graphical entities as well as
                             the routines which know how to draw, erase,
                             and otherwise deal with these.)
  (FNS * AIPSREALIZATIONINITFNS)
  (FNS * AIPSREALIZATIONTOLOCATEFNS)
  (ADDVARS (CKLONEFILES AIPSREALIZATION))
  (FNS * AIPSREALIZATIONDERIVATIONFNS)
  (FNS * AIPSREALIZATIONTOMAKEFNS)
  (FNS * AIPSREALIZATIONTOSIZEFNS)))
```

[DECLARE: DONTEVAL@LOAD DONTCOPY

```
(* The AIPS Realization Model resides here.
This contains the lowest-level description of drawable graphical
entities as well as the routines which know how to draw, erase,
and otherwise deal with these.)
```

}

```
(RPAQQ AIPSREALIZATIONINITFNS (InitCIRCLE InitCLOSEDCURVE InitCURVE
                               InitDISPLAYATOM
                               InitDISPLAYCOMPOSITE
                               InitDISPLAYFORM
                               InitDisplayFormConcepts
                               InitEDGESET InitELLIPSE
                               InitLINESEGMENT InitPOINT
                               InitPOLYGON InitRECTANGLE
                               InitREGULARPOLYGON
                               InitRealizationModel))
```

(DEFINEQ

{1}

```
(InitCIRCLE
  [LAMBDA NIL
```

```
(* Edited by J.Gibbons on
8-Nov-80.)
```

```
  [concept CIRCLE
    (specializes ELLIPSE)
    [roleset Radius
      (mods SemiMinorAxis @ ELLIPSE)
      (mods SemiMajorAxis @ ELLIPSE)]
    (itags (ToDraw TDCircle))]])
```

{2}

```
(InitCLOSEDCURVE
  [LAMBDA NIL
```

```
(* Edited by J.Gibbons on
31-Jul-80.)
```

```
  [concept CLOSEDCURVE
    (specializes CURVE)
```

```
(* May differ from CURVE by the addition of tagged procedures
for determining whether a given point is within the
Approximation (which is expected to be an edgeset that closes
on itself.))
```

```
]])
```

{3}

```
(InitCURVE
  [LAMBDA NIL
```

```
(* Edited by J.Gibbons on
8-Jul-80.)
```

```
  [concept CURVE
```

```
(* We cannot really handle
curves so we approximate them
with an Edgeset.)
```

```
    (specializes DISPLAYATOM)
    [roleset Approximation
      (vr EDGESET)]
    (itags (ToDraw TDCurve))]])
```

{4}

```
(InitDISPLAYATOM
  [LAMBDA NIL
```

```
(* Edited by J.Gibbons on
13-Jan-81.)
```

```
  [concept DISPLAYATOM
```

(* A Display Atom is the most primitive type of Display Form.
It has no constituents and is the final descriptive level
before descent into LISP drawing procedures.)

```
(specializes DISPLAYFORM)
(itags (ToErase TEDisplayAtom)
      (ToLocate TLDisplayAtom)))]])
```

{5}

```
(InitDISPLAYCOMPOSITE
[LAMBDA NIL
```

(* Edited by J.Gibbons on
13-Jan-81.)

```
[concept DISPLAYCOMPOSITE
```

(* A possibly recursive concept
to aggregate DisplayAtoms.)

```
(specializes DISPLAYFORM)
[roleset Component
  (modality Obligatory)
  (number (1 NIL))
  (vr DISPLAYFORM)]
[roleset CoordSys
  (vr CARTESIANSYSTEM)]
```

(* The coordinate system by which the locations of the
component Display Forms are interpreted.)

```
(itags (ToDraw TDDisplayComposite)
      (ToErase TEDisplayComposite)))]])
```

{6}

```
(InitDISPLAYFORM
[LAMBDA NIL
```

(* Edited by J.Gibbons on
13-Jan-81.)

```
[concept DISPLAYFORM
```

(* Display Items come in two flavors: Displays
(graphic Presentations) and Display Forms
(graphic phenomena.))

```
(specializes DISPLAYITEM)]])
```


{7}

```
(InitDisplayFormConcepts
  [LAMBDA NIL
```

```
(* Edited by J.Gibbons on
27-Oct-80.)
```

```
  (InitDISPLAYFORM)
  (InitDISPLAYATOM)
  (InitDISPLAYCOMPOSITE)
  (InitPOINT)
  (InitEDGESET)
  (InitCURVE)
  (InitLINESEGMENT)
  (InitCLOSEDCURVE)
  (InitELLIPSE)
  (InitPOLYGON)
  (InitCIRCLE)
  (InitREGULARPOLYGON)
  (InitRECTANGLE))
```

{8}

```
(InitEDGESET
  [LAMBDA NIL
```

```
(* Edited by J.Gibbons on
8-Jul-80.)
```

```
  [concept EDGESET
```

```
(* A cheap way of representing
a bunch of little contiguous
lines.)
```

```
    (specializes DISPLAYATOM)
    [roleset VertexList
      (modality Obligatory)
      (vr LISTP)]
    (itags (ToDraw TEdgeSet)))]])
```

{9}

```
(InitELLIPSE
  [LAMBDA NIL
```

```
(* Edited by J.Gibbons on
13-Jan-81.)
```

```
  [concept ELLIPSE
    (specializes CLOSEDCURVE)
    [roleset Center
      (mods Location@CLOSEDCURVE)]
    [roleset SemiMinorAxis
      (diffs Size@CLOSEDCURVE)
      (modality Obligatory)]
    [roleset SemiMajorAxis
      (diffs Size@CLOSEDCURVE)
      (modality Obligatory)]
```

```
(itags (ToDraw TDEllipse)))]])
```

{10}

```
(InitLINESEGMENT
[LAMBDA NIL
```

```
(* Edited by J.Gibbons on
8-Nov-80.)
```

```
[concept LINESEGMENT
```

```
(* a finite piece of a line.)
```

```
(specializes CURVE)
```

```
[roleset EndPoint
```

```
(mods Location@CURVE)
```

```
(number 2)]
```

```
[roleset Line
```

```
(vr LINE)
```

```
(derivation (DeriveLineEquationFromEndpoints (Prerequisites
EndPoint)))]])
```

```
(itags (ToDraw TDLineSegment)))]])
```

{11}

```
(InitPOINT
[LAMBDA NIL
```

```
(* Edited by J.Gibbons on
8-Nov-80.)
```

```
[concept POINT
```

```
(specializes DISPLAYATOM)
```

```
(* The Location of a Point is
given in a list of
coordinates.)
```

```
(itags (ToDraw TDPoint)))]])
```

{12}

```
(InitPOLYGON
[LAMBDA NIL
```

```
(* Edited by J.Gibbons on
8-Nov-80.)
```

```
[concept POLYGON
```

```
(specializes DISPLAYCOMPOSITE CLOSEDCURVE)
```

```
[roleset Order
```

```
(vr NUMBERP)]
```

```
[roleset Side
```

```
(mods Component@DISPLAYCOMPOSITE)
```

```
(number (3 NIL))
```

```
(vr LINESEGMENT)]
```

```
(* Note that there is a possible redundancy in how the
positions of the vertices are expressed.
```

There may be an Approximation Edgeset, or the positions may be roles of the Sides.)

11)

{13}

(InitRECTANGLE
[LAMBDA NIL

(* Edited by J.Gibbons on
19-Dec-80.)

```
[concept RECTANGLE
  (specializes POLYGON)
  [roleset NIL
    (mods Order@POLYGON)
    (vr 4)]
  [roleset NIL
    (mods Side@POLYGON)
    (modality Optional)
    (number 4)]
  [roleset LowerLeft
    (diffs Location@POLYGON)
    (defaultfiller '(0 0))]
  [roleset UpperRight
    (diffs Location@POLYGON)
    (derivation (Copy (Prerequisites LowerLeft Height Width)
      (Binding (lowerLeft {$$CONCEPT;LowerLeft}:1))
      (Arguments <lowerLeft:1+({$$CONCEPT;Width}:1-1)
        lowerLeft:2+({$$CONCEPT;Height}:1-1)
        >))))]
  [roleset NIL
    (mods Height@POLYGON)
    [derivation (Copy (Prerequisites LowerLeft UpperRight)
      (Arguments (1+{$$CONCEPT;UpperRight}:1:2
        -{$$CONCEPT;LowerLeft}:1:2))]
  [roleset NIL
    (mods Width@POLYGON)
    [derivation (Copy (Prerequisites LowerLeft UpperRight)
      (Arguments (1+{$$CONCEPT;UpperRight}:1:1
        -{$$CONCEPT;LowerLeft}:1:1))]
  (itags (ToDraw TDRectangle)
    (ToLocate TLRectangle)
    (ToMake TMRectangle)
    (ToSize TSRectangle)))]
```

{14}

(InitREGULARPOLYGON
[LAMBDA NIL

(* Edited by J.Gibbons on
13-Jan-81.)

```
[concept REGULARPOLYGON
  (specializes POLYGON)
  [roleset Center
    (mods Location@POLYGON)]
  [roleset Radius
    (mods Size@POLYGON)]
  [roleset NIL
    (mods Orientation@POLYGON)]
  (itags (ToDraw TDRregularPolygon))]]
```

{15}

```
(InitRealizationModel
[LAMBDA NIL
```

```
(* Edited by J.Gibbons on
21-Jun-80.)
```

```
(* Initializes the AIPS Realization Model.
Concepts at this level have to do with geometric shapes
(which are isomorphic to descriptions of line drawings.))
```

```
(InitDisplayFormConcepts)
```

```
(* Eventually we will init the
bmg stuff here)
```

```
NIL))
```

```
)
```

```
(RPAQQ AIPSREALIZATIONTOLOCATEFNS (TLDisplayAtom TLRectangle))
(DEFINEQ
```

{16}

```
(TLDisplayAtom
[DLAMBDA ((displayAtom IndividualConcept (SATISFIES displayAtom df
`DISPLAYATOM))
  (location (LST OF NUMBERP)))
```

```
(* Edited by J.Gibbons on
29-Jan-81.)
```

```
(* TLDisplayAtom sets the Location Roleset of displayAtom to
location which defaults according to the Location's
DefaultFiller ITag.)
```

```
(if location
  then (SetRoleValues displayAtom Location@DISPLAYATOM <location>)
  else location_{displayAtom;&Location}:1)
location))
```

{17}

```

(TLRectangle
  [DLAMBDA ((rectangle IndividualConcept (SATISFIES rectangle df
                                             ^RECTANGLE))
            (location (LST OF SMALLP)))
    (* Edited by J.Gibbons on
      21-Dec-80.)
    (if location
      then (SetRoleValues rectangle LowerLeft@RECTANGLE <location>)
      else location {rectangle;&LowerLeft}:1)
    (SetRoleValues rectangle UpperRight@RECTANGLE
      <<location:1+({rectangle;&Width}:1-1)
        location:2+({rectangle;&Height}:1-1)
      >>)
    location))
)

(ADDTOVAR CKLONEFILES AIPSREALIZATION)

(RPAQQ AIPSREALIZATIONDERIVATIONFNS (MakeRectangle))
(DEFINEQ

```

{18}

```

(MakeRectangle
  [DLAMBDA ((height NUMBERP)
            (width NUMBERP))
    (* Make up a Rectangle from
      Height and Width information.)
    <(Make RECTANGLE Height height Width width)
      >])
)

(RPAQQ AIPSREALIZATIONTOMAKEFNS (TMRectangle))
(DEFINEQ

```

{19}

```

(TMRectangle
  [LAMBDA (genericRectangle)
    (* Edited by J.Gibbons on
      18-Dec-80.)
    <(Make RECTANGLE)
      >])
)

(RPAQQ AIPSREALIZATIONTOSIZEFNS (TSRectangle))
(DEFINEQ

```

{20}

```
(TSRectangle
[DLAMBDA ((rectangle IndividualConcept (SATISFIES rectangle df
                                         `RECTANGLE))
          (height (ONEOF NIL SMALLP))
          (width (ONEOF NIL SMALLP)))
          (* Edited by J.Gibbons on
            19-Dec-80.)
  (if height
    then (SetRoleValues rectangle Height@RECTANGLE <height>)
    else height_{rectangle;&Height}:1)
  (if width
    then (SetRoleValues rectangle Width@RECTANGLE <width>)
    else width_{rectangle;&Width}:1)
  <height width>])
)
STOP
```

12. SOURCE FILE: AIPSTABLE

FindColumnTableRows.....	10
FindColumnWidth.....	11
FindRowHeight.....	12
FindRowTableColumns.....	13
FindTableEntityIndicators.....	14
FindTableThemeElementIndicators.....	15
FindTableThemeElementIndicatorsAndOrientation.....	16
InitARRANGEMENT.....	1
InitCOLUMN.....	2
InitCOLUMNTable.....	3
InitROW.....	4
InitROWTable.....	5
InitTABLE.....	6
InitTableConcepts.....	9
InitTEXTCOLUMN.....	7
InitTEXTROW.....	8
MakeDefaultTableGround.....	17
MakeTableColumnOrderList.....	18
MakeTableEntryColumns.....	19
MakeTableEntryRows.....	20
MakeTableFoot?.....	21
MakeTableHeaderRow.....	22
MakeTableLeftIndex.....	23
MakeTableRightIndex?.....	24
MakeTableRowOrderList.....	25
TLColumnTable.....	26
TLRowTable.....	27

```
(FILECREATED "18-Dec-80 08:03:51" <NEWAIPS>AIPSTABLE..36 29721
```

```
  changes to: FindColumnTableRows
```

```
  previous date: "17-Dec-80 20:06:11" <NEWAIPS>AIPSTABLE..35)
```

```
(PRETTYCOMPRINT AIPSTABLECOMS)
```

```
(RPAQQ AIPSTABLECOMS ((FNS * AIPSTABLEINITFNS)
                       (FNS * AIPSTABLEDERIVATIONFNS)
                       (FNS * AIPSTABLETOLOCATEFNS)
                       (ADDVARS (CKLONEFILES AIPSTABLE))))
```

```
(RPAQQ AIPSTABLEINITFNS (InitARRANGEMENT InitCOLUMN InitCOLUMNTABLE InitROW
                          InitROWTABLE InitTABLE
                          InitTEXTCOLUMN InitTEXTROW
                          InitTableConcepts))
```

```
(DEFINEQ
```

```
{1}
```

```
(InitARRANGEMENT
[LAMBDA NIL
```

```
(* Edited by F.Zdybel on
18-Jul-80.)
```

```
[concept ARRANGEMENT
```

```
(* An Arrangement is any collection of Display Items whether
or not their arrangement conveys information.
Rows, Columns, Legends are examples of Arrangements.)
```

```
(specializes DISPLAY)
[roleset Item
 (diffs Realization@DISPLAY)
 (modality Obligatory)
 (number (2 NIL))]
```

```
(* If there aren't at least two
items, why bother?)
```

```
[roleset ItemOrder
 (vr LISTP)]
```

```
(* The list which fills this Role can be used to impose an
ordering on the fillers of the Item Role.)
```



```
[roleset Border
  (diffs Realization@DISPLAY)
  (vr CLOSEDCURVE)]
[roleset Separator
  (diffs Realization@DISPLAY)
  (modality Optional)
  (number (0 NIL))
  (vr DISPLAYITEM)]
```

(* Could be anything but is
probably a Line.)))]

{2}

```
(InitCOLUMN
  [LAMBDA NIL
```

(* Edited by F.Zdybel on
12-Dec-80.)

```
    [concept COLUMN
```

(* In contrast to Rows, Columns
are expected to have a linear
structure.)

```
      (specializes ARRANGEMENT)
```

```
      [roleset NIL
        (mods Item@ARRANGEMENT)
        (modality Obligatory)
        (number (2 NIL))
        (vr DISPLAY)]
```

```
      [roleset NIL
        (mods Width@DISPLAYITEM)
        (derivation (FindColumnWidth (Prerequisites Item)
                                         (Arguments {$$CONCEPT;Item})))))]
```

{3}

```
(InitCOLUMNTABLE
  [LAMBDA NIL
```

(* Edited by F.Zdybel on
16-Dec-80.)

```
    [concept COLUMNTABLE
```

(* A Table whose Theme is spread along its columns.
Hence, a row of columns.)

```
      (specializes TABLE ROW)
```

```
      [roleset NIL
        (mods Ground@TABLE)
        (derivation (MakeDefaultTableGround))]
      [roleset NIL
        (mods ThemeList@TABLE)
```

```

(derivation (FindTableThemeElementIndicators (Prerequisites
                                                Application)
                                                (Arguments
                                                  { $$CONCEPT; Application; })))

(PROG [(column [roleset Column
                (mods Group@TABLE Item@ROW)
                (vr TEXTCOLUMN)]]
 [roleset LeftIndex
  (diffs (atomval column))
  (modality Obligatory)
  [derivation (MakeTableLeftIndex (Arguments
                                    { $$CONCEPT; &ThemeList; :1
                                    (CAR
                                      { $$CONCEPT; &Ground; })
                                  })
  (fillwhenmade)]]

(* The LeftIndex is the column of display labels that runs
down the left edge of the ColumnTable.)

[roleset RightIndex
  (diffs (atomval column))
  [derivation (MakeTableRightIndex? (Arguments
                                      (FLENGTH (CAR
                                                  { $$CONCEPT; &EntityList; })
                                                (CAR
                                                  { $$CONCEPT; &LeftIndex; })
                                      ))

(* A very wide Table might also want an index along its right
edge.)

[roleset Entry
  (diffs (atomval column))
  (modality Obligatory)
  (number (1 NIL))
  [derivation (MakeTableEntryColumns (Prerequisites Application)
                                      (Arguments (CAR
                                                  { $$CONCEPT; &EntityList; })
                                                  (CAR
                                                  { $$CONCEPT; &ThemeList; })
                                                  (CAR
                                                  { $$CONCEPT; &Ground; })
                                      ))
  (fillwhenmade)]]
[roleset NIL
  (mods ItemOrder@ARRANGEMENT)
  (derivation (MakeTableColumnOrderList (Arguments (CAR
                                                      { $$CONCEPT; &LeftIndex; })
                                                      { $$CONCEPT; &Entry; } (
                                                        NLSETQ
                                                        { $$CONCEPT; &RightIndex; } :1))

```

```

    ))
  [roleset Row
    (vr TEXTROW)
    [derivation (FindColumnTableRows (Prerequisites Column)
      (Arguments
        { $$CONCEPT; &ItemOrder } : 1
        (FLENGTH
        { $$CONCEPT; ThemeList } : 1))

```

(* While the derivation and realization of a ColumnTable are in terms of its constituent Columns, for layout purposes it is also necessary to describe the table in terms of rows.)

```

  (itags (ToLocate TLColumnTable)
    (NumEntriesJustifyingARightIndex 5)
    (HorizontalSeparation 15)
    (VerticalSeparation 8)))]])

```

{4}

```

(InitROW
  [LAMBDA NIL

```

(* Edited by F.Zdybel on
12-Dec-80.)

```

  [concept ROW

```

(* Note that a Row in a Table may have a non-linear structure (e.g., in order to "compress" the width of the table by putting two interleaved rows together to make a single group in the table.))

```

  (specializes ARRANGEMENT)
  [roleset NIL
    (mods Item@ARRANGEMENT)
    (modality Obligatory)
    (number (2 NIL))
    (vr DISPLAY)]
  [roleset NIL
    (mods Height@DISPLAYITEM)
    (derivation (FindRowHeight (Prerequisites Item)
      (Arguments { $$CONCEPT; Item })))))]])

```

{5}

```

(InitROWTABLE
  [LAMBDA NIL

```

(* Edited by F.Zdybel on

16-Dec-80.)

[concept ROWTABLE

(* A Table whose Theme is spread along its rows.
Hence, a column of rows.)

```
(specializes TABLE COLUMN)
[roleset NIL
  (mods Ground@TABLE)
  (derivation (MakeDefaultTableGround)))]
[roleset NIL
  (mods ThemeList@TABLE)
  (derivation (FindTableThemeElementIndicators (Prerequisites
                                                    Application)
                                                    (Arguments
                                                     {$$CONCEPT;Application}))))]
(PROG [(row [roleset Row
              (mods Group@TABLE Item@COLUMN)
              (vr TEXTROW)])]
  [roleset Header
    (diffs (atomval row))
    (modality Obligatory)
    (derivation (MakeTableHeaderRow (Arguments
                                      {$$CONCEPT;&ThemeList}:1
                                      {$$CONCEPT;&Ground}:1)))
    (fillwhenmade)])
```

(* The Header is the row of display labels that runs across
the top of the RowTable.)

```
[roleset Foot
  (diffs (atomval row))
  (derivation (MakeTableFoot? (Arguments (FLENGTH
                                           {$$CONCEPT;&EntityList}:1
                                           {$$CONCEPT;&Header}:1))
    )]
```

(* A very tall table might want another index running along
the bottom edge.)

```
[roleset Entry
  (diffs (atomval row))
  (modality Obligatory)
  (number (1 NIL))
  (derivation (MakeTableEntryRows (Arguments
                                    {$$CONCEPT;&EntityList}:1
                                    {$$CONCEPT;&ThemeList}:1
```

```

                                { $$CONCEPT;&Ground}:1)))
        (fillwhenmade))]
[roleset NIL
  (mods ItemOrder@ARRANGEMENT)
  (derivation (MakeTableRowOrderList (Arguments { $$CONCEPT;&Header}:1
                                                { $$CONCEPT;&Entry} (
                                                  NLSETQ
                                                  { $$CONCEPT;&Foot}:1)))
              )])
[roleset Column
  (vr TEXTCOLUMN)
  (derivation (FindRowTableColumns (Prerequisites Row)
                                    (Arguments
                                     { $$CONCEPT;&ItemOrder}:1
                                     (FLENGTH
                                      { $$CONCEPT;&ThemeList}:1)))
              )])

```

(* While the derivation and realization of a RowTable are in terms of its constituent Rows, for layout purposes it is also necessary to describe the table in terms of columns.)

```

(itags (ToLocate TLRowTable)
      (NumEntriesJustifyingAFoot 10)
      (HorizontalSeparation 10)
      (VerticalSeparation 10)))]])

```

{6}

```

(InitTABLE
  [LAMBDA NIL

```

```

    [concept TABLE

```

```

      (specializes ARRANGEMENT)
      [roleset NIL
        (mods Application@DISPLAY)
        (number 1)]

```

(* Edited by F.Zdybel on
4-Dec-80.)

(* A two-dimensional table is
conceived of as an Arrangement
of Arrangements.)

(* For the moment we will restrict the Application Role of
table to take a single ITemplate as filler, and the derivation
procedures will enforce this restriction via their DECL, even
if KL-ONE per se does not.)

```

[roleset Group

```

```
(mods Item@ARRANGEMENT)
(vr ARRANGEMENT)]
```

(* Groups are either the columns or rows of the Table.
Just which depends on how one wishes to describe the Table: as
a set of columns or a set of rows.)

```
[roleset EntityList
  (vr LISTP)
  (derivation (FindTableEntityIndicators (Prerequisites Application)
                                           (Arguments
                                            { $$CONCEPT; Application })))])

[roleset ThemeList
  (vr LISTP)
  (derivation (FindTableThemeElementIndicatorsAndOrientation
                (Prerequisites Application)
                (Arguments { $$CONCEPT; Application
                             $$CONCEPT; &EntityList :1 })))
  (fillwhenmade)]
```

(* A list of the highest level slots (domain world Roles)
indicated in the ITemplates filling the application Role.
The ordering corresponds to the ordering of the entries in
each group of the eventual Table.)

]])

{7}

```
(InitTEXTCOLUMN
[LAMBDA NIL
```

(* Edited by F.Zdybel on
25-Jul-80.)

```
[concept TEXTCOLUMN
```

(* A Column whose items are
Texts.)

```
(specializes COLUMN)
[roleset NIL
  (mods Item@COLUMN)
  (vr TEXT)]]])
```

{8}

```
(InitTEXTROW
[LAMBDA NIL
```

(* Edited by F.Zdybel on
25-Jul-80.)

```
[concept TEXTROW
```

(* A Row whose items are
Texts.)

```
(specializes ROW)
[roleset NIL
 (mods Item@ROW)
 (vr TEXT)]]])
```

{9}

```
(InitTableConcepts
 [LAMBDA NIL
```

(* Edited by F.Zdybel on
25-Jul-80.)
(* Initializes Concepts having
to do with Tables.)

```
(InitARRANGEMENT)
(InitROW)
(InitCOLUMN)
(InitTEXTROW)
(InitTEXTCOLUMN)
(InitTABLE)
(InitROWTABLE)
(InitCOLUMNTABLE])
)
```

```
(RPAQQ AIPSTABLEDERIVATIONFNS (FindColumnTableRows FindColumnWidth
                                FindRowHeight
                                FindRowTableColumns
                                FindTableEntityIndicators
                                FindTableThemeElementIndicators
                                FindTableThemeElementIndicatorsAndOrientation
                                MakeDefaultTableGround
                                MakeTableColumnOrderList
                                MakeTableEntryColumns
                                MakeTableEntryRows
                                MakeTableFoot?
                                MakeTableHeaderRow
                                MakeTableLeftIndex
                                MakeTableRightIndex?
                                MakeTableRowOrderList))
```

```
(DEFINEQ
```

{10}

```
(FindColumnTableRows
 [DLAMBDA ((columnList (LST OF IndividualConcept
                        (SATISFIES (HomogenousConceptListP columnList
                                ^COLUMN))))
 (numRows FIXP)
 (RETURNS (LST OF IndividualConcept (SATISFIES
```

```
(HomogenousConceptListP VALUE:1 `ROW))))
(* Edited by F.Zdybel on
18-Dec-80.)
```

```
(* Figures out the rows of a table that has been built in
terms of columns.)
```

```
(for rowNum from 1 to numRows bind itemOrderList
  collect (Make ROW Item itemOrderList_ (for columnDescr
                                          in columnList
                                          collect (FNTH
{columnDescr;ItemOrder}:1
rowNum):1)
ItemOrder itemOrderList)))
```

{11}

```
(FindColumnWidth
[DLAMBDA ((itemList (LST OF IndividualConcept
                    (SATISFIES (HomogenousConceptListP itemList
                                `DISPLAY))))
  (RETURNS (LST OF FIXP)))
(* Edited by F.Zdybel on
12-Dec-80.)
```

```
(* Determines the width of a column of Displays by finding the
maximum width of the items.)
```

```
<(APPLY 'MAX {itemList;&Width})
>])
```

{12}

```
(FindRowHeight
[DLAMBDA ((itemList (LST OF IndividualConcept
                    (SATISFIES (HomogenousConceptListP itemList
                                `DISPLAY))))
  (RETURNS (LST OF FIXP)))
(* Edited by F.Zdybel on
12-Dec-80.)
```

```
(* Determines the height of a row of Displays by finding the
maximum height of the items.)
```

```
<(APPLY 'MAX {itemList;&Height})
>])
```


{13}

```
(FindRowTableColumns
  [DLAMBDA ((rowList (LST OF IndividualConcept (SATISFIES
    (HomogenousConceptListP rowList `ROW))))
    (numColumns FIXP)
    (RETURNS (LST OF IndividualConcept
      (SATISFIES (HomogenousConceptListP VALUE:1
        `COLUMN))))))
  (* Edited by F.Zdybel on
    12-Dec-80.)
```

(* Figures out the columns of a table that has been built in terms of rows.)

```
(for columnNum from 1 to numColumns bind itemOrderList
  collect (Make COLUMN Item itemOrderList_ (for rowDescr in rowList
    collect (FNTH
      {rowDescr;ItemOrder}:1
        columnNum):1)
    ItemOrder itemOrderList)))
```

{14}

```
(FindTableEntityIndicators
  [DLAMBDA ((tableApplicationList (LST OF IndividualConcept) (SATISFIES
    (
      tableApplicationList:1 df `ITEMPLATE and ~tableApplicationList:1)))
    (RETURNS (LST OF (LST OF IndividualConcept)) (SATISFIES
      (
        HomogenousConceptListP VALUE:1 `CONCEPT))))
    (* Edited by F.Zdybel on
      4-Dec-80.)
    (* Gets the ConceptGroup
      fillers of the template filling
      the Application Role.)
    < {tableApplicationList:1;ConceptGroup} >])
```

{15}

```
(FindTableThemeElementIndicators
  [DLAMBDA ((tableApplicationList (LST OF IndividualConcept)
    (SATISFIES tableApplicationList:1 df `ITEMPLATE
      and ~tableApplicationList:1)))
    (RETURNS (LST OF (LST OF IndividualConcept))
      (SATISFIES HomogenousConceptListP VALUE:1 `ROLE))))
```

(* Edited by F.Zdybel on
4-Dec-80.)

(* Gathers together the roles meta-indicated by the ITemplate
filling the Application Role. Gives special treatment to any
Role name Name, by placing it first in the list.)

```
(PROG ((themeRoleIndicatorList {tableApplicationList:1;RoleGroup})
  nameRoleIndicator)
  (if nameRoleIndicator_ [for roleIndicator
    in themeRoleIndicatorList
    thereis
    ('Name)
    fmemb
    (KLFindNamesOfRole
    (KLGetMetaDescribedEntity
    roleIndicator
    (KLGetNamedContext
    'AIPSDOMAINMETA]
    then themeRoleIndicatorList
    <nameRoleIndicator ! (DREMOVE nameRoleIndicator
    themeRoleIndicatorList)
    >)
  (RETURN < themeRoleIndicatorList >)))])
```

{16}

```
(FindTableThemeElementIndicatorsAndOrientation
[DLAMBDA ((tableApplicationList (LST OF IndividualConcept)
  (SATISFIES (tableApplicationList:1 df `ITEMPLATE
    and ~tableApplicationList::1)))
  (tableEntityIndicatorList (LST OF IndividualConcept)
  (SATISFIES HomogenousConceptListP
    tableEntityIndicatorList `CONCEPT)))
  (RETURNS [LST OF (LST OF IndividualConcept
    (SATISFIES (HomogenousConceptListP
    VALUE:1
    `ROLE]))))

(* Edited by F.Zdybel on
4-Dec-80.)
```

(* Gathers together the role meta-indicators on the ITemplate
filling the Application Role. Gives special treatment to the
indicator of any Role named Name, by placing it first in the
list. Finally compares the number of entities in the table
with the number of theme elements and decides whether the
Table should become a RowTable or a ColumnTable.)

```

(PROG ((themeRoleIndicatorList {tableApplicationList:1;RoleGroup})
  nameRoleIndicator temp)
  (if nameRoleIndicator_ [for roleIndicator
                        in themeRoleIndicatorList
                        thereis
                        ('Name)
                        fmem
                        (KLFindNamesOfRole
                        (KLGetMetaDescribedEntity
                        roleIndicator
                        (KLGetNamedContext
                        'AIPSDOMAINMETA]
    then themeRoleIndicatorList
      <nameRoleIndicator ! (DREMOVE nameRoleIndicator
                                themeRoleIndicatorList)
      >)
    (if (FLENGTH tableEntityIndicatorList) lt temp (FLENGTH
                                                themeRoleIndicatorList)
      and temp gt 3
      then $$PROTOTYPE `COLUMNTABLE
      else $$PROTOTYPE `ROWTABLE)
    (RETURN < themeRoleIndicatorList >)))

```

{17}

```

(MakeDefaultTableGround
  [DLAMBDA ((RETURNS (LST OF IndividualConcept) (SATISFIES (VALUE:1 df
                                                            `VIEWSURFACE))))
    (* Edited by F.Zdybel on
    13-Nov-80.)

```

(* Make a ViewSurface of random size so that at least all of
the Displays in the Table can have their Ground roles filled.
The dimensions of the ViewSurface can be changed later.)

```

<(Make VIEWSURFACE Continuum (Make CARTESIANSYSTEM)
  Boundary
  (Make RECTANGLE LowerLeft <0 0> UpperRight <100 100>))
>])

```

{18}

```

(MakeTableColumnOrderList
  [DLAMBDA ((leftIndex IndividualConcept
              (SATISFIES (leftIndex df `COLUMN)))
    (entryColumnList (LST OF IndividualConcept)
      (SATISFIES HomogenousConceptListP entryColumnList
                  `COLUMN)))
    (rightIndex (ONEOF (IndividualConcept

```

```

                (SATISFIES (rightIndex df `COLUMN)))
            NIL))
    (RETURNS (LST OF (LST OF IndividualConcept))
      (SATISFIES HomogenousConceptListP VALUE:1 `COLUMN)))
      (* Edited by F.Zdybel on
        13-Nov-80.)

```

(* Produces the list that imposes the correct ordering on the Columns of the ColumnTable. Hopefully entryColumnList already reflects the ordering embedded in the ColumnTable's Application.)

```

(PROG ((result (<leftIndex ! < ! entryColumnList>>)))
  (if rightIndex
    then result < !! result rightIndex>)
  (RETURN <result>)))

```

{19}

```

(MakeTableEntryColumns
[DLAMBDA ((tableEntityList (LST OF IndividualConcept)
  (SATISFIES (HomogenousConceptListP tableEntityList
    `CONCEPT)))
  (tableThemeList (LST OF IndividualConcept)
    (SATISFIES (HomogenousConceptListP tableThemeList
      `ROLE)))
  (tableGround IndividualConcept (SATISFIES tableGround df
    `VIEWSURFACE))
  (RETURNS (LST OF IndividualConcept) (SATISFIES
    (HomogenousConceptListP
      VALUE `COLUMN))))
  (* Edited by F.Zdybel on
    4-Dec-80.)

```

(* Produces the Columns in a ColumnTable;
one Column for each element of EntityList.
Each Column contains one Item for each element in ThemeList.
Fills the ItemOrder Role of the Table to indicate the ordering
of the Columns, and fills the ItemOrder Role of each Column to
indicate the ordering of the items in the Column.)

```

(for entityIndicator in tableEntityList
  bind itemOrder (itemPrototype _ (KLGetConceptName (KLFindVRsOfRole
    Item@{ $$PROTOTYPE;Entry}:1):1))
  collect (Make TEXTCOLUMN Application
    (Make ITEMPLATE ConceptGroup entityIndicator RoleGroup
      tableThemeList)

```

```

Item itemOrder_(for themeElementIndicator
                in tableThemeList
                collect (APPLY* 'Make itemPrototype
                              'Application
                              (Make ITEMPLATE
                               ConceptGroup
                               entityIndicator
                               RoleGroup
                               themeElementIndicator)
                              'Ground tableGround))
ItemOrder itemOrder Ground tableGround)))

```

{20}

```

(MakeTableEntryRows
 [DLAMBDA ((tableEntityList (LST OF IndividualConcept)
                             (SATISFIES (HomogenousConceptListP tableEntityList
                                     `CONCEPT)))
          (tableThemeList (LST OF IndividualConcept)
                           (SATISFIES (HomogenousConceptListP tableThemeList
                                   `ROLE)))
          (tableGround IndividualConcept (SATISFIES tableGround df
                                     `VIEWSURFACE))
          (RETURNS (LST OF IndividualConcept) (SATISFIES
                                                (HomogenousConceptListP
                                                 VALUE `ROW))))
 (* Edited by F.Zdybel on
  4-Dec-80.)

```

(* Produces the Rows in a RowTable; one Row for each element of EntityList. Each Row contains one Item for each element in ThemeList. Fills the ItemOrder Role of the Table to indicate the ordering of the Rows, and fills the ItemOrder Role of each Row to indicate the ordering of the items in the Row.)

```

(for entityIndicator in tableEntityList
  bind itemOrder (itemPrototype _(KLGetConceptName (KLFindVRsOfRole
                                                    Item@{ $$PROTOTYPE;Entry}:1):1))
  collect (Make TEXTROW Application (Make ITEMPLATE ConceptGroup
                                                    entityIndicator RoleGroup
                                                    tableThemeList)
          Item itemOrder_(for themeElementIndicator
                          in tableThemeList
                          collect (APPLY* 'Make itemPrototype
                                        'Application
                                        (Make ITEMPLATE
                                         ConceptGroup
                                         entityIndicator

```

```

                                RoleGroup
                                themeElementIndicator)
                                'Ground tableGround))
ItemOrder itemOrder Ground tableGround))]]

```

{21}

```

(MakeTableFoot?
[DLAMBDA ((numberOfRows NUMBERP)
  (tableHeader IndividualConcept
    (SATISFIES tableHeader df `ROW))
  (RETURNS (LST OF IndividualConcept)
    (SATISFIES VALUE:1 df `ROW)))
  (* Edited by F.Zdybel on
  13-Nov-80.)

(* Decides whether or not to create a Foot for a RowTable by
comparing the numberOfRows with the value of the
NumEntriesJustifyingAFoot tag. The Foot, if created, is a copy
of the Header.)

(PROG ((tableGround (CAR {tableHeader;Ground})))
  (if [IGEQ numberOfRows (CAR (KLFindIData $$CONCEPT '(
    NumEntriesJustifyingAFoot]
    then (RETURN
      <[Make TEXTROW Ground tableGround Item
        (for headerItem in {tableHeader;Item}
          collect (Make TEXT Ground tableGround String
            (CAR {headerItem;String}
              >)]
    else (StopFill))))))

```

{22}

```

(MakeTableHeaderRow
[DLAMBDA ((tableThemeList (LST OF IndividualConcept)
  (SATISFIES (HomogenousConceptListP tableThemeList
    `ROLE)))
  (tableGround IndividualConcept (SATISFIES tableGround df
    `VIEWSURFACE))
  (RETURNS (LST OF IndividualConcept)
    (SATISFIES VALUE:1 df `ROW)))
  (* Edited by F.Zdybel on
  7-Dec-80.)

```

(* Derives the Header Row for a RowTable, by going through the Roles meta-indicated by the elements of themeList and taking the most specific names of those roles.)

```

<(Make TEXTROW Item itemOrder_(for roleMetaIndicator
    in tableThemeList
    collect
    (Make
    TEXT String
    [MKSTRING
    (CAR
    (KLFindNamesOfRole
    (KLGetMetaDescribedEntity
    roleMetaIndicator
    (KLGetNamedContext
    'AIPSDOMAINMETA]
    Ground tableGround))
    ItemOrder itemOrder Ground tableGround)
>])

```

{ 23 }

```

(MakeTableLeftIndex
[DLAMBDA ((tableThemeList (LST OF IndividualConcept)
    (SATISFIES (HomogenousConceptListP tableThemeList
    `ROLE)))
    (tableGround IndividualConcept (SATISFIES tableGround df
    `VIEWSURFACE))
    (RETURNS (LST OF IndividualConcept) (SATISFIES VALUE:1 df
    `COLUMN)))
    (* Edited by F.Zdybel on
    7-Dec-80.)

```

(* Derives the LeftIndex for a ColumnTable, by going through the Roles meta-indicated by the elements of themeList and taking the most specific names of those roles.)

```

<(Make TEXTCOLUMN Item itemOrder_(for roleMetaIndicator
    in tableThemeList
    collect
    (Make
    TEXT String
    [MKSTRING
    (CAR
    (KLFindNamesOfRole
    (KLGetMetaDescribedEntity
    roleMetaIndicator
    (KLGetNamedContext
    'AIPSDOMAINMETA]
    Ground tableGround))
    ItemOrder itemOrder Ground tableGround)

```

>])

{ 24 }

```
(MakeTableRightIndex?
[DLAMBDA ((numberOfColumns NUMBERP)
  (tableLeftIndex IndividualConcept (SATISFIES tableLeftIndex df
                                           ^COLUMN))
  (RETURNS (LST OF IndividualConcept) (SATISFIES VALUE:1 df
                                           ^COLUMN)))
  (* Edited by F.Zdybel on
  13-Nov-80.)
```

(* Decides whether or not to create a RightIndex for a ColumnTable by comparing the numberOfColumns with the value of the NumEntriesJustifyingARightIndex tag.
The RightIndex, if created, is a copy of the LeftIndex.)

```
(PROG ((tableGround (CAR {tableLeftIndex;Ground})))
  (if [IGEQ numberOfColumns (CAR (KLFindIData $$CONCEPT '(
    NumEntriesJustifyingARightIndex]
    then (RETURN
      <[Make TEXTCOLUMN Ground tableGround Item
        (for leftIndexItem in {tableLeftIndex;Item}
          collect (Make TEXT Ground tableGround String
                    (CAR {leftIndexItem;String})
        >)
    else (StopFill))))))
```

{ 25 }

```
(MakeTableRowOrderList
[DLAMBDA ((headerRow IndividualConcept (SATISFIES (headerRow df ^ROW)))
  (entryRowList (LST OF IndividualConcept)
    (SATISFIES (HomogenousConceptListP entryRowList ^ROW)))
  (footRow (ONEOF (IndividualConcept
    (SATISFIES (footRow df ^ROW)))
    NIL))
  (RETURNS (LST OF (LST OF IndividualConcept))
    (SATISFIES (HomogenousConceptListP VALUE:1 ^ROW))))
  (* Edited by F.Zdybel on
  13-Nov-80.)
```

(* Produces the list that imposes the correct ordering on the Rows of the RowTable. Hopefully entryRowList already reflects the ordering embedded in the RowTable's Application.)


```

(PROG ((result (<headerRow ! < ! entryRowList>>)))
  (if footRow
    then result_ < !! result footRow>)
  (RETURN <result_>)))
)

(RPAQQ AIPSTABLETOLOCATEFNS (TLColumnTable TLRowTable))
(DEFINEQ

```

{ 26 }

```

(TLColumnTable
  [DLAMBDA ((tableDescr IndividualConcept (SATISFIES tableDescr df
    ^COLUMNNTABLE))
    (tableLoc (ONEOF NIL (LST OF FIXP (SATISFIES ~tableLoc::2))))))
    (* Edited by F.Zdybel on
    17-Dec-80.)

(* Locates the interior elements of a ColumnTable, given the
location of the ColumnTable's lower left corner.
If no location is given, the location
(0,0) is assumed)

(if ~tableLoc
  then tableLoc '(0 0))
(for columnDescr in (CAR {tableDescr;&ItemOrder}))
  bind (currentX _ tableLoc:1)
  currentY _
  (bottomY _ tableLoc:2)
  [vertSeparation _ (CAR (KLFindIData tableDescr '(
    VerticalSeparation]
  [horizSeparation _ (CAR (KLFindIData tableDescr '(
    HorizontalSeparation]
  (rowHeightList _ {tableDescr;&Row;&Height}))
  do (currentY _ bottomY)
  (SendMessage columnDescr 'ToLocate <currentX currentY>)
  (for itemDescr in (REVERSE (CAR {columnDescr;&ItemOrder})))
    as rowHeight in (REVERSE rowHeightList)
    do (SendMessage itemDescr 'ToLocate <currentX currentY>)
      (currentY _ currentY+rowHeight+vertSeparation))
  (currentX _ currentX+(CAR {columnDescr;&Width})+horizSeparation)
  finally (KLChangeRoleValue (KLFindOneNamedInstanceRole (CAR
    {tableDescr;Ground;Boundary}))
    'UpperRight)
    <currentX currentY>)))]))

```

{ 27 }

```
(TLRowTable
[DLAMBDA ((tableDescr IndividualConcept (SATISFIES (tableDescr df
                                                    ^ROWTABLE)))
          (tableLoc (ONEOF NIL (LST OF FIXP (SATISFIES ~tableLoc::2))))))
          (* Edited by F.Zdybel on
            17-Dec-80.)
```

(* Locates the interior elements of a RowTable, given the location of the RowTable's lower left corner. If no location is given, the location (0,0) is assumed.)

```
(if ~tableLoc
    then tableLoc '(0 0))
(for rowDescr in (REVERSE {tableDescr;&ItemOrder}:1)
  bind (leftX _ tableLoc:1)
        currentX
        (currentY _ tableLoc:2)
        [vertSeparation _ (CAR (KLFindIData tableDescr '(
                                                              VerticalSeparation)
                                [horizSeparation _ (CAR (KLFindIData tableDescr '(
                                                              HorizontalSeparation]
                                (columnWidthList _ {tableDescr;&Column;&Width}))
  do currentX leftX
    (SendMessage rowDescr 'ToLocate <currentX currentY>)
    (for itemDescr in {rowDescr;ItemOrder}:1 as columnWidth
      in columnWidthList do (SendMessage itemDescr 'ToLocate
                                                <currentX currentY>)
                            currentX
                            currentX+columnWidth+horizSeparation)
    currentY currentY+{rowDescr;&Height}:1+vertSeparation
  finally (KLChangeRoleValue (KLFindOneNamedInstanceRole (CAR
                                                            {tableDescr;Ground;Boundary}))
                            'UpperRight)
          <currentX currentY>))))
)

(ADDTovar CKLONEFILES AIPSTABLE)
STOP
```

13. SOURCE FILE: AIPSTEMP

FindMostSpecializedConcept.....	17
HomogenousConceptListP.....	18
InitAIPS.....	1
InitDISPLAYITEM.....	2
InitDISPLAYITEMABSTRACTION.....	3
InitDISPLAYSURFACE.....	4
InitHOMOGENEOUSREGION.....	5
InitINVISIBLESURFACE.....	6
InitMiddleEarth.....	8
InitMOTILEDISPLAYITEM.....	7
InitPLANESURFACE.....	9
InitRECTANGULARREGION.....	10
InitSTABILEDISPLAYITEM.....	11
InitUSERMODEL.....	12
InitUserModel.....	13
InitViewingOrganizationModel....	16
InitVIEWSURFACE.....	14
InitVISIBLESURFACE.....	15
TLDisplayItem.....	19
TLDisplaySurface.....	20
TLPlaneSurface.....	21
TMDisplaySurface.....	22
TMRegionNumber.....	23
TSPlaneSurface.....	24
TSRectangularRegion.....	25

(FILECREATED " 2-Feb-81 03:43:55" <NEWAIPS>AIPSTEMP..111 18702

changes to: InitDISPLAYSURFACE AIPSTEMPCOMS AIPSTEMPUTILITYFNS
AIPSTEMPTOMAKEFNS TMRegionNumber

previous date: "31-Jan-81 04:23:25" <NEWAIPS>AIPSTEMP..110)

(PRETTYCOMPRINT AIPSTEMPCOMS)

```
(RPAQQ AIPSTEMPCOMS [(FNS * AIPSTEMPINITFNS)
  (FNS * AIPSTEMPUTILITYFNS)
  (FNS * AIPSTEMPTOLOCATEFNS)
  (FNS * AIPSTEMPTOMAKEFNS)
  (FNS * AIPSTEMPTOSIZEFNS)
  (DECLARE: DONTEVAL@LOAD EVAL@COMPILEWHEN
    (NOT (BOUNDP (QUOTE BMGCOMS)))
    DONTCOPY
    (FILES <FONTWORK>BMGRECORDSANDVARS..0))
  (ADDVARS (CKLONEFILES AIPSTEMP))
  (DECLARE: DONTEVAL@LOAD DOEVAL@COMPILE DONTCOPY
    COMPILERVARS (ADDVARS (NLAMA)
      (NLAML)
      (LAMA])
```

```
(RPAQQ AIPSTEMPINITFNS (InitAIPS InitDISPLAYITEM InitDISPLAYITEMABSTRACTION
  InitDISPLAYSURFACE InitHOMOGENEOUSREGION
  InitINVISIBLESURFACE InitMOTILEDISPLAYITEM
  InitMiddleEarth InitPLANESURFACE
  InitRECTANGULARREGION
  InitSTABILEDISPLAYITEM InitUSERMODEL
  InitUserModel InitVIEWSURFACE
  InitVISIBLESURFACE
  InitViewingOrganizationModel))
```

(DEFINEQ

{1}

```
(InitAIPS
[LAMBDA NIL
```

```
(* Edited by J.Gibbons on
11-Nov-80.)
(* Initializes the
Presentation, Viewing
Organization and Realization
Models of AIPS.)
```

```
(InitDISPLAYITEM)
(InitDISPLAYITEMABSTRACTION)
(InitUserModel)
```

```

(InitCoordinateSystemConcepts)
(InitTimeSystemConcepts)
(InitCommonDomainModel)
(InitNavalDomainModel)
(InitRealizationModel)
(InitPresentationModel)
(InitNTDSConcepts)
(InitViewingOrganizationModel)
(InitDeviceConcepts])

```

[2]
[2]
[2]

```

(InitDISPLAYITEM
  [LAMBDA NIL

```

(* Edited by J.Gibbons on
20-Jan-81.)

```

    [concept DISPLAYITEM

```

(* A Display Item can be either a Presentation or a Display Form. Thus decomposition of Presentations occurs until the level of Display Forms is reached.)

```

    [roleset Location
      (vr LISTP)
      (defaultfiller <0 0>)]

```

(* Relates the location of the origin of the abstract figure (in the case of Displays and DisplayComposites) or is further differentiated or modified in the case of DisplayAtoms (e.g.: the EndPoint Role of LINE.))

```

    [roleset Size
      (vr NUMBERP)
      (defaultfiller 1)]

```

(* Indicates a scaling transformation which is applied to the abstract figure before translation and after rotation (in the case of Displays and DisplayComposites) or is further modified or differentiated in the case of DisplayAtoms (e.g.: the SemiMajorAxis and SemiMinorAxis Roles of ELLIPSE.) The filler is always a positive number greater than or equal to one since we do not ever wish to scale abstract figures down, and in the case of DisplayAtoms we are specifying size information in terms of pixels.)

```
[roleset Orientation
  (vr NUMBERP)
  (defaultfiller 0)]
```

(* Indicates a rotational transformation which is applied to the abstract figure before translation and scaling (in the case of Displays and DisplayComposites) or is further modified in the case of DisplayAtoms (e.g.: the Orientation Role of ELLIPSE.) The filler is expected to indicate in terms of a number of degrees between -180 and 180 inclusive.)

```
[roleset Width
  (vr NUMBERP)]
[roleset Height
  (vr NUMBERP)]
```

(* These Roles are used by layout processes that operate at the level of the Viewing Organization Model.)

```
[roleset Envelope
  (vr CLOSEDCURVE)]
```

(* This role is useful for characterizing the area subsumed by a Display Item.)

```
(itags (ToLocate TLDisplayItem)))]])
```

[3]
[3]
[3]

```
(InitDISPLAYITEMABSTRACTION
[LAMBDA NIL
```

(* Edited by J.Gibbons on
11-Nov-80.)

```
[concept DISPLAYITEMABSTRACTION
```

(* Abstractions of DisplayItems
are individuals of this
concept.)

```
(specializes ABSTRACTION)
[roleset FixedComponent
  (number (1 NIL))
  (vr DISPLAYFORM)]
```

(* Any visible components of the DisplayItem that are fixed in terms of identity and location (e.g.: the circle and dot of an NTDS friendly ship symbol.))

(itags (ToDraw TDDisplayItemAbstraction))]])

[4]
[4]
[4]

(InitDISPLAYSURFACE
[LAMBDA NIL

(* Edited by J.Gibbons on
2-Feb-81.)

[concept DISPLAYSURFACE

(* A Viewsurface that represents an actual chunk of bitmap memory, either visible or invisible.)

(specializes DISPLAYATOM VIEWSURFACE)
[roleset Plane
(modality Obligatory)
(vr PLANESURFACE)]

(* Indicates which of the primal Display Surfaces the Display Surface is located on.)

[roleset NIL
(mods Height@DISPLAYATOM)
(mods Height@VIEWSURFACE)]
[roleset NIL
(mods Width@DISPLAYATOM)
(mods Width@VIEWSURFACE)]
[roleset RegionNumber
(vr NUMBERP)
(derivation (TMRegionNumber (Prerequisites Plane Location Height
Width Background)
(Arguments { \$\$CONCEPT;Plane}:1
\$\$CONCEPT;Location}:1
\$\$CONCEPT;Height}:1
\$\$CONCEPT;Width}:1
\$\$CONCEPT;Background}:1)))]

(* Establishes the correspondence with a BMG Display Region more directly than via Nexii.)

```
[roleset Background
(modality Obligatory)
(vr SMALLP)
(defaultfiller 0)]
```

(* Sets the Background shade of
the DisplaySurface.)

```
[roleset NIL
(mods Continuum@VIEWSURFACE)
(vr VIEWSYSTEM)]
```

(* Because a DISPLAYSURFACE represents actual map memory, the
coordinate system must be integer Cartesian.)

```
(itags (ToDraw TDDisplaySurface)
(ToLocate TLDisplaySurface)
(ToMake TMDisplaySurface)))]
```

[5]
[5]
[5]

```
(InitHOMOGENEOUSREGION
[LAMBDA NIL
```

(* Edited by J.Gibbons on
5-Sep-80.)

```
[concept HOMOGENEOUSREGION
(specializes REGION)
[roleset NIL
(mods Aperture)
(number 0)]]])
```

[6]
[6]
[6]

```
(InitINVISIBLESURFACE
[LAMBDA NIL
```

(* Edited by J.Gibbons on
5-Sep-80.)

```
[concept INVISIBLESURFACE
```

(* Here the medium is constrained to be a Display Region onto
one of the off-screen planes of bitmap memory.)

```
(specializes DISPLAYSURFACE)
[roleset NIL
(mods Plane@DISPLAYSURFACE)
(vr [iconcept HIDDENPLANE of PLANESURFACE])
```



```
(itags (DefaultFiller `HIDDENPLANE))]]])
```

[7]
[7]
[7]

```
(InitMOTILEDISPLAYITEM  
[LAMBDA NIL
```

```
(* Edited by J.Gibbons on  
5-Sep-80.)
```

```
[concept MOTILEDISPLAYITEM
```

```
(* The purpose of this concept is to mark which Display Items  
can be re-located (those whose locations are not significant  
to their semantics.))
```

```
(specializes DISPLAYITEM))])
```

[8]
[8]
[8]

```
(InitMiddleEarth  
[LAMBDA NIL
```

```
(* Edited by F.Zdybel on  
31-Jan-80.)
```

```
(* Initializes AIPS and the necessary Domain World Model  
SI-Net fragments for the Middle Earth demonstration system.)
```

```
(InitAIPS)  
(InitGeographyModel])
```

[9]
[9]
[9]

```
(InitPLANESURFACE  
[LAMBDA NIL
```

```
(* Edited by J.Gibbons on  
31-Jan-81.)
```

```
[concept PLANESURFACE
```

```
(* Represents a plane of BMG-11  
bitmap memory.)
```

```
(specializes DISPLAYSURFACE)  
[roleset PlaneNumber  
  (modality Obligatory)  
  (vr NUMBERP)]  
[roleset NIL  
  (mods Location@DISPLAYSURFACE)  
  (derivation (Copy (Prerequisites PlaneNumber))
```

```

        (Binding (specListRecord (BMGShowPlane
                                { $$CONCEPT;PlaneNumber}:1)))
        (Arguments <specListRecord:6 specListRecord:10>)))
]
[roleset NIL
 (mods Height@DISPLAYSURFACE)
 (derivation (Copy (Prerequisites PlaneNumber)
                  (Binding (specListRecord (BMGShowPlane
                                { $$CONCEPT;PlaneNumber}:1)))
                  (Arguments 1+specListRecord:12-specListRecord:10)))
)]
[roleset NIL
 (mods Width@DISPLAYSURFACE)
 (derivation (Copy (Prerequisites PlaneNumber)
                  (Binding (specListRecord (BMGShowPlane
                                { $$CONCEPT;PlaneNumber}:1)))
                  (Arguments 1+specListRecord:8-specListRecord:6))))
 (itags (ToLocate TLPlaneSurface)
        (ToSize TSPlaneSurface))]]

```

[10]
[10]
[10]

(InitRECTANGULARREGION
[LAMBDA NIL

(* Edited by J.Gibbons on
19-Dec-80.)

```

[concept RECTANGULARREGION
 (specializes HOMOGENEOUSREGION)
 [roleset NIL
  (mods Boundary@HOMOGENEOUSREGION)
  (modality Obligatory)
  (vr RECTANGLE)
  (derivation (TMRectangle))]
 [roleset NIL
  (mods Continuum@HOMOGENEOUSREGION)
  (vr CARTESIANSYSTEM)]
 [roleset NIL
  (mods Height@HOMOGENEOUSREGION)
  (derivation (Copy (Prerequisites Boundary)
                    (Arguments { $$CONCEPT;Boundary;Height}:1))))]
 [roleset NIL
  (mods Width@HOMOGENEOUSREGION)
  (derivation (Copy (Prerequisites Boundary)
                    (Arguments { $$CONCEPT;Boundary;Width}:1))))]
 (itags (ToSize TSRectangularRegion))]]

```

[11]
[11]
[11]

```
(InitSTABILEDISPLAYITEM
  [LAMBDA NIL
```

```
(* Edited by J.Gibbons on
5-Sep-80.)
```

```
[concept STABILEDISPLAYITEM
```

```
(* The purpose of this concept is to mark which display items
cannot be re-located (those whose locations are significant to
their semantics.))
```

```
(specializes DISPLAYITEM)]])
```

```
[12]
[12]
[12]
```

```
(InitUSERMODEL
  [LAMBDA NIL
```

```
(* Edited by J.Gibbons on
13-Nov-80.)
```

```
(* A simple collection of parameters reflecting some of a
user's preferences regarding system behavior.)
```

```
[concept USERMODEL
  [roleset LabelLocality
    (fillwhenmade)
    (vr ATOM)
    (defaultfiller 'TOP)]
  [roleset ApertureHeight
    (fillwhenmade)
    (vr SMALLP)
    (defaultfiller 256)]
  [roleset ApertureWidth
    (fillwhenmade)
    (vr SMALLP)
    (defaultfiller 256)]])
```

```
[13]
[13]
[13]
```

```
(InitUserModel.
  [LAMBDA NIL
```

```
(* Edited by J.Gibbons on
24-Sep-80.)
(* Defines the UserModel
concept and instantiates a
default model.)
```

```
(InitUSERMODEL)
(AIPSUSERMODEL_(Make USERMODEL))
```

[14]

[14]

[14]

(InitVIEWSURFACE

[LAMBDA NIL

[concept VIEWSURFACE

(specializes RECTANGULARREGION)

[roleset Window

(number (0 NIL))

(vr WINDOW)]

(* Edited by J.Gibbons on
5-Sep-80.)(* A View Surface is a medium
for the arrangement of viewable
objects.)(* An aspect of View Surfaces
is that Windows open onto
them.))])

[15]

[15]

[15]

(InitVISIBLESURFACE

[LAMBDA NIL

[concept VISIBLESURFACE

(* Here the medium is constrained to be a Display Region onto
one of the visible planes. This difference is reflected in a
different derivation of the Display Region filling the Medium
Role.)

(specializes DISPLAYSURFACE)

[roleset NIL

(mods Plane@DISPLAYSURFACE)

(itags (DefaultFiller "CONRACSCREEN"))])])

(* Edited by J.Gibbons on
5-Sep-80.)

[16]

[16]

[16]

(InitViewingOrganizationModel

[LAMBDA NIL

(* Edited by J.Gibbons on
5-Sep-80.)(* Initializes the Viewing Organization Model of the AIPS
SI-Net. These concepts have to do with the syntactic

organization of display elements, and include such as Window, View Surface, Display Form, etc.)

```
(InitMOTILEDISPLAYITEM)
(InitSTABILEDISPLAYITEM)
```

(* Set up all the necessary concepts for describing positions on View Surfaces, Display Surfaces, etc.)

```
(InitHOMOGENEOUSREGION)
(InitRECTANGULARREGION)
(InitVIEWSURFACE)
(InitDISPLAYSURFACE)
(InitVISIBLESURFACE)
(InitINVISIBLESURFACE)
(InitPLANESURFACE)
```

(* Sets up the generic descriptions of the different types of Windows.)

```
(InitWindowConcepts])
```

```
)
```

```
(RPAQQ AIPSTEMPUTILITYFNS (FindMostSpecializedConcept
                           HomogenousConceptListP))
(DEFINEQ
```

[17]
[17]
[17]

```
(FindMostSpecializedConcept
 [DLAMBDA ((conceptList (LST OF GenericConcept))
           (RETURNS GenericConcept))
```

(* Edited by F.Zdybel on 3-Jul-80.)

(* Tries to find the concept that is at the lowest level of the hierarchy. Picks the first concept on conceptList that has no ancestors on conceptList.)

```
(for concept in conceptList thereis (for otherConcept in conceptList
                                     when (concept~=otherConcept)
                                     never (otherConcept df concept)))
```

```
])
```

[18]
[18]
[18]

```

(HomogenousConceptListP
[LAMBDA (cList restrictionC)
  (for element in cList always element df restrictionC))
)

(RPAQQ AIPSTEMPTOLOCATEFNS (TLDisplayItem TLDisplaySurface TLPlaneSurface))
(DEFINEQ

```

[19]
[19]
[19]

```

(TLDisplayItem
[DLAMBDA ((displayItemDescr IndividualConcept (SATISFIES
  (displayItemDescr df
    'DISPLAYITEM)))
  (itemLoc (LISTP OF FIXP (SATISFIES ~itemLoc::2))))
  (* Edited by F.Zdybel on
  16-Dec-80.)
  (* Simply locates the
  DisplayItem at the specified
  location.)

(PROG (currentLocationRole)
  (if currentLocationRole_ (KLFindOneNamedInstanceRole
    displayItemDescr
    'Location)
    then (KLChangeRoleValue currentLocationRole itemLoc)
    else (KLSatisfyRole Location@DISPLAYITEM displayItemDescr
      <itemLoc>))))))

```

[20]
[20]
[20]

```

(TLDisplaySurface
[DLAMBDA ((displaySurface IndividualConcept (SATISFIES displaySurface df
  'DISPLAYSURFACE))
  (location (LST OF FIXP)))
  (* Edited by J.Gibbons on
  31-Jan-81.)
  (if location or ~location_{displaySurface;Location}:1
    then (if location
      then (SetRoleValues displaySurface Location@DISPLAYSURFACE
        <location>)
        (PROG ((regionNumber {displaySurface;RegionNumber}:1))
          (if regionNumber
            then (BMGFreeRegionNumber regionNumber)
            (SetRoleValues displaySurface
              RegionNumber@DISPLAYSURFACE)))
        else location {displaySurface;&Location}:1)
      (SendMessage {displaySurface;&Boundary}:1 'ToLocate location))

```

location])

[21]
[21]
[21]

```
(TLPlaneSurface
  [DLAMBDA ((planeSurface IndividualConcept (SATISFIES planeSurface df
                                                    \PLANESURFACE)))
    (* Edited by J.Gibbons on
    29-Jan-81.)
    (* The location cannot be set
    from outside but it can be
    derived.)
    (TLDisplaySurface planeSurface {planeSurface;&Location}:1)))
)
```

```
(RPAQQ AIPSTEMPTOMAKEFNS (TMDisplaySurface TMRegionNumber))
(DEFINEQ
```

[22]
[22]
[22]

```
(TMDisplaySurface
  [LAMBDA (genericDisplaySurface plane background)
    (* Edited by J.Gibbons on
    19-Dec-80.)
    <(Make DISPLAYSURFACE Plane plane Background background Boundary 'Fill)
    >])
```

[23]
[23]
[23]

```
(TMRegionNumber
  [DLAMBDA ((planeSurface IndividualConcept (SATISFIES planeSurface df
                                                    \PLANESURFACE))
    (location (LISTP OF FIXP))
    (height FIXP)
    (width FIXP)
    (background FIXP)
    (RETURNS (LISTP OF BMGRegionNumber)))
    (* Edited by J.Gibbons on
    2-Feb-81.)
```

(* Initializes a BMGRegion on the specified BMGPlane according
to boundary and returns its BMGRegionNumber.)

<(BMGDefineRegion

```

    <'PLANE {planeSurface;&PlaneNumber}:1 'BACKGROUND background 'MINX
      location:1 'MINY location:2 'MAXX location:1+width-1 'MAXY
      location:2+height-1>)
  >])
)

(RPAQQ AIPSTEMPTOSIZEFNS (TSPlaneSurface TSRectangularRegion))
(DEFINEQ

[24]
[24]
[24]

(TSPlaneSurface
[DLAMBDA ((planeSurface IndividualConcept (SATISFIES planeSurface df
  PLANE SURFACE)))
  (* Edited by J.Gibbons on
    28-Jan-81.)

(* The size of the PlaneSurface cannot be set from outside but
it can be derived from the related BMGRegion.)

[PROG (height width)
  (if height_{planeSurface;Height}:1 and
    width_{planeSurface;Width}:1
    then (RETURN <height width>)
    else (RETURN (TSRectangularRegion planeSurface
      {planeSurface;&Height}:1
      {planeSurface;&Width}:1]))

[25]
[25]
[25]

(TSRectangularRegion
[DLAMBDA ((rectangularRegion IndividualConcept (SATISFIES
  rectangularRegion df
  RECTANGULAR REGION))
  (height (ONEOF NIL FIXP))
  (width (ONEOF NIL FIXP))
  (* Edited by J.Gibbons on
    28-Jan-81.)
  (if height or width or ~(height_{rectangularRegion;Height}:1
    and width_{rectangularRegion;Width}:1)
    then (if height
      then (SetRoleValues rectangularRegion
        Height@RECTANGULAR REGION <height>)
      else height_{rectangularRegion;&Height}:1)
    (if width
      then (SetRoleValues rectangularRegion

```



```

                                Width@RECTANGULARREGION <width>)
      else width_{rectangularRegion;&Width}:1)
      (SendMessage {rectangularRegion;&Boundary}:1 'ToSize height
                                width))
      <height width>])
)
(DECLARE: DONTEVAL@LOAD EVAL@COMPILEWHEN (NOT (BOUNDP (QUOTE BMGCOMS)))
DONTCOPY
(FILELOAD <FONTWORK>BMGRECORDSANDVARS..0)
)

(ADDTOTVAR CKLONEFILES AIPSTEMP)
STOP

```


14. SOURCE FILE: AIPSTODRAW

DisplaySurfaceForGroundP.....	27
FindDrawingContextsForDisplay.....	28
FindRoleValuesTransitively.....	29
FindTopLevelDisplaysOnViewSurface.....	30
GetFontCharacterDef.....	31
LDIFFERENCE.....	32
RotatePointsAroundPoint.....	34
ROUND.....	33
RoundPoint.....	35
RoundPoints.....	36
ScrollFastWindow.....	1
ScrollSlowWindow.....	2
ScrollTTYWindow.....	3
ScrollWindow.....	37
TDCircle.....	4
TDCurve.....	5
TDDisplay.....	6
TDDisplayComposite.....	7
TDDisplayItemAbstraction.....	8
TDDisplaySurface.....	9
TDEdgeSet.....	10
TDEllipse.....	11
TDFastWindow.....	12
TDIWFastWindow.....	13
TDIWNonScrollWindow.....	14
TDIWSlowWindow.....	15
TDLineSegment.....	16
TDNonScrollWindow.....	17
TDPoint.....	18
TDRectangle.....	19
TDRegularPolygon.....	20
TDSlowWindow.....	21
TDText.....	23
TDTTYWindow.....	22
TDWindow.....	24
TEDisplayAtom.....	25
TEDisplayComposite.....	26
TransformPoints.....	38
TransformSize.....	39

(FILECREATED " 2-Feb-81 03:12:47" <NEWAIPS>AIPSTODRAW..63 38495

changes to: ScrollFastWindow

previous date: " 2-Feb-81 02:48:19" <NEWAIPS>AIPSTODRAW..62)

(PRETTYCOMPRINT AIPSTODRAWCOMS)

(RPAQQ AIPSTODRAWCOMS ((* The following are the ToDraw functions. They are called only after all locating has been done.)

(FNS * AIPSTODRAWFNS)

(FNS * AIPSTODRAWERASEFNS)

(DECLARE: COPYWHEN (NOT COMPILEIGNOREDECL)
(RECORDS * AIPSTODRAWRECORDS))

(FNS * AIPSTODRAWUTILITYFNS)

(GLOBALVARS TDRRegion)

(DECLARE: DONTEVAL@LOAD EVAL@COMPILEWHEN
(NOT (BOUNDP (QUOTE BMGCOMS)))
DONTCOPY

(FILES <FONTWORK>BMGRECORDSANDVARS..0))
(ADDVARS (CKLONEFILES AIPSTODRAW)))

[DECLARE: DONTEVAL@LOAD DONTCOPY

(* The following are the ToDraw functions.
They are called only after all locating has been done.)

]

(RPAQQ AIPSTODRAWFNS (ScrollFastWindow ScrollSlowWindow ScrollTTYWindow
TDCircle TDCurve TDDisplay
TDDisplayComposite
TDDisplayItemAbstraction
TDDisplaySurface TDEdgeSet TDEllipse
TDFastWindow TDIWFastWindow
TDIWNonScrollWindow TDIWSlowWindow
TDLineSegment TDNonScrollWindow
TDPoint TDRectangle TDRegularPolygon
TDSlowWindow TDTTYWindow TDText
TDWindow))

(DEFINEQ

{1}

```

(ScrollFastWindow
  [DLAMBDA ((fastWindow IndividualConcept (SATISFIES fastWindow df
    `FASTWINDOW))
    location)
    (CLISP: MIXED)
    (* Edited by J.Gibbons on
    2-Feb-81.)

    (* Redraws the aperture and its contents according to the new
    location or the old substrate location if no new one is
    given.)

    (PROG ((apertureDisplaySurface {fastWindow;&Aperture;&DisplaySurface}:1)
      (substrate {fastWindow;&Substrate}:1)
      locationOfSubstrate substrateRegion)
      (if location
        then (SetRoleValues fastWindow SubstrateLocation@FASTWINDOW
          <location>)
        else location_ {fastWindow;&SubstrateLocation}:1)
      (locationOfSubstrate_ {substrate;&Location}:1)
      (substrateRegion_ (TMRegionNumber `HIDDENPLANE
        <locationOfSubstrate:1+location:1
          locationOfSubstrate:2+location:2>
          {apertureDisplaySurface;&Height}:1
          {apertureDisplaySurface;&Width}:1):1)
      (BMGCopyRegion substrateRegion
        {apertureDisplaySurface;&RegionNumber}:1)
      (BMGFreeRegionNumber substrateRegion)))

    {2}

```

```

(ScrollSlowWindow
  [DLAMBDA ((slowWindow IndividualConcept (SATISFIES slowWindow df
    `SLOWWINDOW))
    location)
    (CLISP: MIXED)
    (* Edited by J.Gibbons on
    27-Jan-81.)

    (* Redraws the aperture and its contents according to the new
    location or the old substrate location if no new one is
    given.)

    (if location
      then (SetRoleValues slowWindow SubstrateLocation@SLOWWINDOW
        <location>)
      else location_ {slowWindow;&SubstrateLocation}:1)
    (location_ (create PointRecord xCoord_-location:1

```

```

        yCoord -location:2))
(bind (apertureDisplaySurface (CAR
                                {slowWindow;&Aperture;&DisplaySurface}))
      first (SendMessage apertureDisplaySurface 'ToDraw) for display
      in (FindTopLevelDisplaysOnViewSurface (CAR {slowWindow;&Substrate}))
      do (SendMessage display 'ToDraw apertureDisplaySurface location)))

```

{ 3 }

```

(ScrollTTYWindow
[DLAMBDA ((ttyWindow IndividualConcept
            (SATISFIES ttyWindow df `TTYWINDOW))
          (height (ONEOF NIL FIXP)))
          (* Edited by J.Gibbons on
            28-Jan-81.)
          (* Scrolls the TTYWindow up
            height pixels or, if NIL, a
            default amount.)
          (BMGScrollRegion height {ttyWindow;&Substrate;&RegionNumber}:1))

```

{ 4 }

```

(TDCircle
[DLAMBDA ((circle IndividualConcept (SATISFIES circle df `CIRCLE))
          (displaySurface IndividualConcept (SATISFIES displaySurface df
                                                        `DISPLAYSURFACE))
          (translation (LST OF NUMBERP))
          (magnification (ONEOF NIL NUMBERP))
          (rotationSpec (ONEOF LST NUMBERP))
          (displayMode (ONEOF NIL BMGDisplayMode)))
          (* Edited by J.Gibbons on
            13-Jan-81.)
          (* Draws a circle.)
          (PROG ((regionNumber {displaySurface;&RegionNumber}:1)
                  (center {circle;&Center}:1)
                  (radius {circle;&Radius}:1))
                (center_(create PointRecord xCoord_center:1
                                              yCoord_center:2))
                (center_(RoundPoint (TransformPoints <center> rotationSpec
                                              magnification translation):1)
                )
                (radius_(ROUND (TransformSize radius magnification)))
                (BMGDisplayMode (if displayMode
                                     else 'ADD)
                                 regionNumber)
                (BMGMove center:xCoord center:yCoord regionNumber)
                (BMGCircle radius regionNumber)))

```

{5}

```

(TDCurve
  [DLAMBDA ((curve IndividualConcept (SATISFIES (curve df `CURVE)))
            (displaySurface IndividualConcept (SATISFIES displaySurface df
                                                        `DISPLAYSURFACE))
            (translation (LST OF NUMBERP))
            (magnification (ONEOF NIL NUMBERP))
            (rotationSpec (ONEOF LST NUMBERP))
            (displayMode (ONEOF NIL BMGDisplayMode)))
            (* Edited by J.Gibbons on
              13-Jan-81.)
            (* Draws a curve via its
              Approximation)
  (PROG ((approximation {curve;&Approximation}:1)
        (SendMessage approximation 'ToDraw displaySurface translation
                                   magnification rotationSpec displayMode)))

```

{6}

```

(TDDisplay
  [DLAMBDA ((display IndividualConcept (SATISFIES display df `DISPLAY))
            (drawingContext (ONEOF NIL IndividualConcept) (SATISFIES
                                                              (OR
                                                                ~drawingContext
                                                                (
drawingContext df `DISPLAYSURFACE)
                                                                (
drawingContext df `WINDOW))))
            (translation (LST OF NUMBERP)))
            (* Edited by J.Gibbons on
              19-Jan-81.)
            (* Draws an individual of
              Display when the drawing
              context is not specified.)
  (if drawingContext df `DISPLAYSURFACE
    then [if (DisplaySurfaceForGroundP drawingContext
                                         {display;&Ground}:1)
            then (PROG ((location {display;&Location}:1)
                      (size {display;&Size}:1)
                      (orientation {display;&Orientation}:1))
                  (location_(create PointRecord xCoord_location:1
                                                yCoord_location:2))
                  (location_(TransformPoints <location> NIL NIL
                                             translation):1)
                      (* Draw abstractions.)
                  (bind abstraction for concept
                    in (KLZGetSuperConcepts display)
                    when abstraction_(KLGetAbstraction concept)

```

```

do (SendMessage abstraction 'ToDraw
    drawingContext location size
    orientation))
    (* Draw DisplayForm
    Realizations.)
(bind genericDisplayForm `DISPLAYFORM
    for realization in {display;Realization}
    when realization df genericDisplayForm
    do (SendMessage realization 'ToDraw
        drawingContext translation]
        (* Draw subDisplays.)
(bind genericDisplay `DISPLAY for realization
    in {display;Realization} when realization df genericDisplay
    do (SendMessage realization 'ToDraw drawingContext
        translation))
elseif drawingContext df `WINDOW
    then (SendMessage drawingContext 'ToDrawInWindow display)
elseif ~drawingContext
    then (for drawingContext in (FindDrawingContextsForDisplay display)
        do (SendMessage display 'ToDraw drawingContext))))

```

{7}

```

(TDDisplayComposite
[DLAMBDA ((displayComposite IndividualConcept (SATISFIES
    (displayComposite df
        `DISPLAYCOMPOSITE)))
    (displaySurface IndividualConcept (SATISFIES displaySurface df
        `DISPLAYSURFACE))
    (translation (LST OF NUMBERP))
    (magnification (ONEOF NIL NUMBERP))
    (rotationSpec (MEMQ NIL 0))
    (displayMode (ONEOF NIL BMGDisplayMode)))
    (* Edited by J.Gibbons on
    16-Jan-81.)

```

(* Draws an individual of DisplayComposite.
We cannot handle an incoming rotationSpec but this seems to be
the only such limitation.)

```

(PROG ((components {displayComposite;&Component}))
    (location {displayComposite;&Location}:1)
    (size {displayComposite;&Size}:1)
    (orientation {displayComposite;&Orientation}:1))
(location_(create PointRecord xCoord_location:1
    yCoord_location:2))
(location_(TransformPoints <location> NIL magnification
    translation):1)
(size_(TransformSize size magnification))

```



```

(for component in components
  do (SendMessage component 'ToDraw displaySurface location size
    orientation displayMode))))

```

{8}

```

(TDDisplayItemAbstraction
  [DLAMBDA ((abstraction IndividualConcept (SATISFIES abstraction df
    ^DISPLAYITEMABSTRACTION))
    (displaySurface IndividualConcept (SATISFIES displaySurface df
    ^DISPLAYSURFACE))
    (translation (LST OF NUMBERP))
    (magnification (ONEOF NIL NUMBERP))
    (rotationSpec (ONEOF NIL NUMBERP)))
    (* Edited by J.Gibbons on
    16-Jan-81.)
    (* Draws an individual of
    DisplayItemAbstraction
    according to the parameters
    given.)
    (for component in {abstraction;FixedComponent}
      do (SendMessage component 'ToDraw displaySurface translation
        magnification rotationSpec))))

```

{9}

```

(TDDisplaySurface
  [DLAMBDA ((displaySurface IndividualConcept (SATISFIES displaySurface df
    ^DISPLAYSURFACE))
    (baseDisplaySurface (ONEOF NIL IndividualConcept) (SATISFIES
    ^baseDisplaySurface
    or
    (
    baseDisplaySurface df ^DISPLAYSURFACE)))
    (translation (LST OF NUMBERP))
    (magnification (ONEOF NIL NUMBERP))
    (rotationSpec (ONEOF LST NUMBERP))
    (displayMode (ONEOF NIL BMGDisplayMode)))
    (* Edited by J.Gibbons on
    19-Jan-81.)

```

```

(* Fills the associated BMGRegion with its Background
according to displayMode. The excess formal arguments are to
make the function call compatible with other DisplayForms.)

```

```

(PROG ((regionNumber {displaySurface;&RegionNumber}:1))
  (BMGDisplayMode (if displayMode

```

```

                else 'OVERWRITE)
                regionNumber)
    (BMGFillRegion NIL regionNumber)))]

```

{10}

```

(TDEdgeSet
[DLAMBDA ((edgeSet IndividualConcept (SATISFIES edgeSet df `EDGESET))
        (displaySurface IndividualConcept (SATISFIES displaySurface df
        `DISPLAYSURFACE))

        (translation (LST OF NUMBERP))
        (magnification (ONEOF NIL NUMBERP))
        (rotationSpec (ONEOF LST NUMBERP))
        (displayMode (ONEOF NIL BMGDisplayMode)))

        (* Edited by J.Gibbons on
        9-Jan-81.)

```

(* Draws an Edge Set; ie. a set of line segments determined by
a list of vertices.)

```

(PROG ((regionNumber {displaySurface;&RegionNumber}:1)
      (vertexList {edgeSet;&VertexList}:1))
(vertexList_(for vertex in vertexList
              collect (create PointRecord xCoord_vertex:1
              yCoord_vertex:2)))
(vertexList_(RoundPoints (TransformPoints vertexList rotationSpec
              magnification
              translation)))

(BMGDisplayMode (if displayMode
                    else 'ADD)
                regionNumber)
(BMGPoint vertexList:1:xCoord vertexList:1:yCoord regionNumber)
(BMGPolyLine vertexList regionNumber)))]

```

{11}

```

(TDEllipse
[DLAMBDA ((ellipse IndividualConcept (SATISFIES ellipse df `ELLIPSE))
        (displaySurface IndividualConcept (SATISFIES displaySurface df
        `DISPLAYSURFACE))

        (translation (LST OF NUMBERP))
        (magnification (ONEOF NIL NUMBERP))
        (rotationSpec (ONEOF LST NUMBERP))
        (displayMode (ONEOF NIL BMGDisplayMode)))

        (* Edited by J.Gibbons on
        13-Jan-81.)
        (* Draws an ellipse.)

        (CLISP: MIXED)

        (PROG ((regionNumber {displaySurface;&RegionNumber}:1)

```

```

    (center {ellipse;&Center}:1)
    (semiMajorAxis {ellipse;&SemiMajorAxis}:1)
    (semiMinorAxis {ellipse;&SemiMinorAxis}:1)
    (orientation {ellipse;&Orientation}:1))
  (center_(create PointRecord xCoord_center:1
                             yCoord_center:2))
  (center_(RoundPoint (TransformPoints <center> rotationSpec
                                     magnification translation):1)
    )
  (semiMajorAxis_(ROUND (TransformSize semiMajorAxis magnification))
    )
  (semiMinorAxis_(ROUND (TransformSize semiMinorAxis magnification))
    )
  (if (LISTP rotationSpec)
    then rotationSpec rotationSpec:1)
  (orientation_(ROUND (REMAINDER orientation+(if rotationSpec
                                                  else 0)
                    180)))
  (if orientation gt 90
    then orientation_orientation-180
    elseif orientation lt -90
    then orientation_orientation+180)
  (BMGDisplayMode (if displayMode
                    else 'ADD)
    regionNumber)
  (BMGMove center:xCoord center:yCoord regionNumber)
  (BMGEllipse semiMinorAxis semiMajorAxis orientation regionNumber))
])

```

{12}

```

(TDFastWindow
  [DLAMBDA ((fastWindow IndividualConcept (SATISFIES (fastWindow df
                                                       `FASTWINDOW)))
    drawingContext)

```

(* Edited by J.Gibbons on
2-Feb-81.)

(* Draws a FastWindow in the specified drawing context.
It must locate the substrate on the HIDDENPLANE, draw the
appropriate displays there, and then copy the substrate to the
aperture.)

```

(if drawingContext df `DISPLAYSURFACE
  then (PROG ((substrate {fastWindow;&Substrate}:1))
    (TDWindow fastWindow drawingContext)
    (SendMessage substrate 'ToLocate
      (FindRegion `HIDDENPLANE
        {substrate;&Height}:1

```

```

                                {substrate;&Width}:1))
    (SendMessage substrate 'ToDraw `HIDDENPLANE)
    (for display in (FindTopLevelDisplaysOnViewSurface
                    substrate)
      do (SendMessage display 'ToDraw substrate))
    (SendMessage fastWindow 'Scroll))
elseif drawingContext df `WINDOW
  then (SHOULDNT)
elseif ~drawingContext
  then (for drawingContext in (FindDrawingContextsForDisplay
                              fastWindow)
    do (SendMessage fastWindow 'ToDraw drawingContext))))

```

{13}

```

(TDIWFastWindow
[DLAMBDA ((fastWindow IndividualConcept (SATISFIES fastWindow df
                                                `FASTWINDOW))
  (display IndividualConcept (SATISFIES display df `DISPLAY)))
  (* Edited by J.Gibbons on
    2-Feb-81.)
  (SendMessage display 'ToDraw {fastWindow;&Substrate}:1)
  (SendMessage fastWindow 'Scroll)))

```

{14}

```

(TDIWNonScrollWindow
[DLAMBDA ((nonScrollWindow IndividualConcept (SATISFIES nonScrollWindow
                                                        df
                                                        `NONSCROLLWINDOW))
  (display IndividualConcept (SATISFIES display df `DISPLAY)))
  (* Edited by J.Gibbons on
    27-Jan-81.)
  (* Knows how to draw a Display
    in a NonScrollWindow.)
  (SendMessage display 'ToDraw {nonScrollWindow;&Substrate}:1)))

```

{15}

```

(TDIWSlowWindow
[DLAMBDA ((slowWindow IndividualConcept (SATISFIES slowWindow df
                                                `SLOWWINDOW))
  (display IndividualConcept (SATISFIES display df `DISPLAY)))
  (CLISP: MIXED)
  (* Edited by J.Gibbons on
    27-Jan-81.)
  (* Knows how to draw Displays
    in SlowWindows.)
  (PROG ((substrateLocation {slowWindow;&SubstrateLocation}:1))

```

```
(SendMessage display 'ToDraw
  {slowWindow;&Aperture;&DisplaySurface}:1
  (create PointRecord xCoord_-substrateLocation:1
    yCoord_-substrateLocation:2))))
```

{16}

```
(TDLineSegment
  [DLAMBDA ((lineSegment IndividualConcept (SATISFIES lineSegment df
    ^LINESEGMENT))
    (displaySurface IndividualConcept (SATISFIES displaySurface df
    ^DISPLAYSURFACE))
    (translation (LST OF NUMBERP))
    (magnification (ONEOF NIL NUMBERP))
    (rotationSpec (ONEOF LST NUMBERP))
    (displayMode (ONEOF NIL BMGDisplayMode)))
    (* Edited by J.Gibbons on
      9-Jan-81.)
    (* Draws a line segment.)
  (PROG ((regionNumber {displaySurface;&RegionNumber}:1)
    (endPoints {lineSegment;&EndPoint}))
    (endPoints_ (for point in endPoints
      collect (create PointRecord xCoord_point:1
        yCoord_point:2)))
    (endPoints_ (RoundPoints (TransformPoints endPoints rotationSpec
      magnification
      translation)))
    (BMGDisplayMode (if displayMode
      else 'ADD)
      regionNumber)
    (BMGLine endPoints:1:xCoord endPoints:1:yCoord endPoints:2:xCoord
      endPoints:2:yCoord regionNumber)))
```

{17}

```
(TDNonScrollWindow
  [DLAMBDA ((nonScrollWindow IndividualConcept (SATISFIES nonScrollWindow
    df
    ^NONSCROLLWINDOW))
    drawingContext)
    (* Edited by J.Gibbons on
      27-Jan-81.)
    (* Draws a NonScrollWindow in
      the specified drawing context.)
  (if drawingContext df ^DISPLAYSURFACE
    then (PROG ((substrate {nonScrollWindow;&Substrate}:1)
      (TDWindow nonScrollWindow drawingContext)
      (for display in (FindTopLevelDisplaysOnViewSurface
        substrate)
        do (SendMessage display 'ToDraw substrate)))
```

```

elseif drawingContext df `WINDOW
then (SHOULDNT)
elseif ~drawingContext
then (for drawingContext in (FindDrawingContextsForDisplay
                             nonScrollWindow)
      do (SendMessage nonScrollWindow 'ToDraw drawingContext))))

```

{18}

```

(TDPoint
[DLAMBDA ((point IndividualConcept (SATISFIES point df `POINT))
         (displaySurface IndividualConcept (SATISFIES displaySurface df
                                                    `DISPLAYSURFACE))

         (translation (LST OF NUMBERP))
         (magnification (ONEOF NIL NUMBERP))
         (rotationSpec (ONEOF LST NUMBERP))
         (displayMode (ONEOF NIL BMGDisplayMode)))
        (* Edited by J.Gibbons on
          9-Jan-81.)
        (* Draws a point.)
(PROG ((regionNumber {displaySurface;&RegionNumber}:1)
      (location {point;&Location}:1))
      (location_(create PointRecord xCoord_location:1
                                yCoord_location:2))
      (location_(RoundPoint (TransformPoints <location> rotationSpec
                                magnification translation)
                            :1))
      (BMGDisplayMode (if displayMode
                          else 'ADD)
                      regionNumber)
      (BMGPoint location:xCoord location:yCoord regionNumber))))

```

{19}

```

(TDRectangle
[DLAMBDA ((rectangle IndividualConcept
                    (SATISFIES rectangle df `RECTANGLE))
         (displaySurface IndividualConcept (SATISFIES displaySurface df
                                                    `DISPLAYSURFACE))

         (translation (LST OF NUMBERP))
         (magnification (ONEOF NIL NUMBERP))
         (rotationSpec (ONEOF LST NUMBERP))
         (displayMode (ONEOF NIL BMGDisplayMode)))
        (* Edited by J.Gibbons on
          9-Jan-81.)
(CLISP: MIXED)

(* Draws a rectangle from the lowerLeft-upperRight
representation. Orientation is with respect to the center of
the rectangle.)

```

```

(PROG ((regionNumber {displaySurface;&RegionNumber}:1)
      (lowerLeft {rectangle;&LowerLeft}:1)
      (upperRight {rectangle;&UpperRight}:1)
      (orientation {rectangle;Orientation}:1)
      corners)
      (lowerLeft_(create PointRecord xCoord_lowerLeft:1
                                   yCoord_lowerLeft:2))
      (upperRight_(create PointRecord xCoord_upperRight:1
                                   yCoord_upperRight:2))
      (corners_(RoundPoints (TransformPoints
                            (TransformPoints
                             <(create PointRecord
                                xCoord_lowerLeft:xCoord
                                yCoord_upperRight:yCoord)
                                upperRight
                                (create PointRecord
                                 xCoord_upperRight:xCoord
                                 yCoord_lowerLeft:yCoord)
                                lowerLeft>
                             <orientation (create
                                PointRecord xCoord_ (
                                lowerLeft:xCoord+upperRight:xCoord)
                                /2.0 yCoord_ (
                                lowerLeft:yCoord+upperRight:yCoord)
                                /2.0)
                             >)
                            rotationSpec magnification translation)))
      (BMGDisplayMode (if displayMode
                          else 'ADD)
                      regionNumber)
      (BMGMove corners:4:xCoord corners:4:yCoord regionNumber)
      (BMGPolyLine corners regionNumber)))

```

{20}

```

(TDRegularPolygon
 [DLAMBDA ((regularPolygon IndividualConcept (SATISFIES
                                             (regularPolygon df
                                              ^REGULARPOLYGON))))
  (CLISP: MIXED)
  (* Edited by J.Gibbons on
  31-Jan-81.)
  (* Draws a regular polygon
  knowing its Center,
  Orientation, Radius, and
  Order.)
  [PROG ((center {regularPolygon;Center}:1)
        (radius {regularPolygon;Radius}:1)
        (order {regularPolygon;Order}:1)

```

```

(orientation (OR {regularPolygon;Orientation}:1 0))
vertex vertexList)
(if center=NIL or radius=NIL or order=NIL
  then
    (* insufficient information for
    drawing)
    (TDFail 'TDRegularPolygon regularPolygon)
  else [vertexList_ (bind (iterationAngle 360./order)
    (listOfPoint <<center:1+radius
    center:2>>)
    from 1 to order as angle from orientation
    by iterationAngle
    collect (vertex (TransformPoint
    (RotatePointsAroundPoint
    < center angle > listOfPoint)
    :1)
    (BMGMove vertex:1 vertex:2)
    (BMGPolyLine (for vertex in vertexList
    collect <vertex:1 ! vertex:2>]]))

```

{ 21 }

```

(TDSlowWindow
[DLAMBDA ((slowWindow IndividualConcept (SATISFIES slowWindow df
`SLOWWINDOW))
drawingContext)
(* Edited by J.Gibbons on
26-Jan-81.)
(* Draws a SlowWindow on the
specified DisplaySurface.)
(if drawingContext df `DISPLAYSURFACE
  then (TDWindow slowWindow drawingContext)
  (SendMessage slowWindow 'Scroll)
elseif drawingContext df `WINDOW
  then (SHOULDNT)
elseif ~drawingContext
  then (for drawingContext in (FindDrawingContextsForDisplay
slowWindow)
do (SendMessage slowWindow 'ToDraw drawingContext)))]])

```

{ 22 }

```

(TDTTYWindow
[DLAMBDA ((ttyWindow IndividualConcept
(SATISFIES ttyWindow df `TTYWINDOW))
drawingContext)
(* Edited by J.Gibbons on
27-Jan-81.)
(* Draws a TTYWindow and ties
the BMGTTYRegion to its

```



```

Substrate.)
(if drawingContext df `DISPLAYSURFACE
  then (TDWindow ttyWindow drawingContext)
        (BMGTTYRegion {ttyWindow;&Substrate;&RegionNumber}:1)
        (BMGResetTTY)
  elseif drawingContext df `WINDOW
  then (SHOULDNT)
  elseif ~drawingContext
  then (for drawingContext in (FindDrawingContextsForDisplay
                                ttyWindow)
      do (SendMessage ttyWindow 'ToDraw drawingContext))))))

{23}

(TDText
  [DLAMBDA ((text IndividualConcept (SATISFIES text df `TEXT))
            (displaySurface IndividualConcept (SATISFIES displaySurface df
                                                         `DISPLAYSURFACE))
            (translation (LST OF NUMBERP))
            (magnification (ONEOF NIL NUMBERP))
            (rotationSpec (ONEOF LST NUMBERP))
            (displayMode (ONEOF NIL BMGDisplayMode)))
    (* Edited by J.Gibbons on
      31-Jan-81.)

  (* Draws the specified text string at the desired location
  using the correct font.)

  [PROG ((fontNumber {text;&Font}:1)
        (lineSpacing -{text;&LineSpacing}:1)
        (location {text;&Location}:1)
        (string {text;&String}:1))
    (* Set the Graphics region and
    its font and starting
    location.)
    (BMGRegion {text;&Ground;&RegionNumber}:1)
    (BMGFont fontNumber)
    (BMGMove location:1 location:2+{text;&Height}:1-(BMGDescribeFont
      fontNumber):LoadedFont\FontRecord:Font\BaseLine)
    (* Output the string.)
    (bind [breakChars_ (CONSTANT (MAKEBITTABLE '(10 13 31]
      lineBeginPos_1
      subString_"" breakChar breakPos
      while breakPos_ (STRPOS breakChars string lineBeginPos)
      do (if subString_ (SUBSTRING string lineBeginPos breakPos-1
        subString)
        then (BMGString subString)
        else subString_ "")
      (lineBeginPos_ breakPos+1)
      (if breakChar_ (CHCON1 (NTHCHAR string breakPos))=31

```

then

(* LISP's internal end of line character.
Treat it like a carriage return, line feed.)

```

      (BMGMove location:1 ((BMGY)+lineSpacing))
elseif breakChar=10
  then (* Bare line feed.)
      (BMGMoveRel (MakeTextWidth subString fontNumber):1
                lineSpacing)
elseif (CHCON1 (NTHCHAR string lineBeginPos))=10
  then (* Carriage return, line feed.)
      (lineBeginPos_lineBeginPos+1)
      (BMGMove location:1 ((BMGY)+lineSpacing))
else (* Bare carriage return.)
      (BMGMove location:1 (BMGY))
finally (if subString_(SUBSTRING string lineBeginPos NIL
                        subString)
        then (BMGString subString]))

```

{24}

```

(TDWindow
[DLAMBDA ((window IndividualConcept (SATISFIES (window df `WINDOW)))
         (drawingContext (ONEOF NIL IndividualConcept) (SATISFIES
         (OR
         ~drawingContext
         (
drawingContext df `DISPLAYSURFACE)
         (
drawingContext df `WINDOW))))
  (translation (LST OF NUMBERP)))
(* Edited by J.Gibbons on
19-Jan-81.)

```

(* Draws the Window as a Display on the specified
DisplaySurface. The parts must be drawn in the proper order;
eg the DisplaySurface which provides the background for the
window as a whole must be drawn first.)

```

(if drawingContext df `DISPLAYSURFACE
  then (if (DisplaySurfaceForGroundP drawingContext
    {window;&Context}:1)
    then (bind drawingSurface {window;&DisplaySurface}:1
      first (SendMessage drawingSurface 'ToDraw)
      for realization in {window;Realization}
      unless realization=drawingSurface

```

```

                                do (SendMessage realization 'ToDraw drawingContext
                                translation))
elseif drawingContext df `WINDOW
  then
                                (* we cannot handle this
                                currently)
                                (SHOULDNT)
elseif ~drawingContext
  then (for drawingContext in (FindDrawingContextsForDisplay window)
        do (SendMessage window 'ToDraw drawingContext)))
)
(RPAQQ AIPSTODRAWERASEFNS (TEDisplayAtom TEDisplayComposite))
(DEFINEQ

```

{25}

```

(TEDisplayAtom
  [LAMBDA (displayAtom displaySurface translation magnification
            rotationSpec)
    (* Edited by J.Gibbons on
    13-Jan-81.)

```

(* Erases the displayAtom by drawing it in FLIP mode.
This works satisfactorily for most DisplayAtoms.
For those where it does not, there will be more specialized
erasing functions.)

```

(SendMessage displayAtom 'ToDraw displaySurface translation
  magnification rotationSpec 'FLIP))

```

{26}

```

(TEDisplayComposite
  [DLAMBDA ((displayComposite IndividualConcept (SATISFIES
            (displayComposite df
              `DISPLAYCOMPOSITE)))
            (displaySurface IndividualConcept (SATISFIES displaySurface df
              `DISPLAYSURFACE))
            (translation (LST OF NUMBERP))
            (magnification (ONEOF NIL NUMBERP))
            (rotationSpec (MEMQ NIL 0)))
    (* Edited by J.Gibbons on
    13-Jan-81.)

```

(* Erases an individual of DisplayComposite.
We cannot handle an incoming rotationSpec but this seems to be
the only such limitation. This is identical to
TDDisplayComposite except that the recursive request is to
erase rather than to draw.)

```

(PROG ((components {displayComposite;&Component}))
      (location {displayComposite;&Location}:1)
      (size {displayComposite;&Size}:1)
      (orientation {displayComposite;&Orientation}:1))
(location_ (TransformPoints <location> NIL magnification
              translation):1)
(size_ (TransformSize size magnification))
(for component in components
  do (SendMessage component 'ToErase displaySurface location
                          size orientation))))
)
(DECLARE: COPYWHEN (NOT COMPILEIGNOREDECL)

(RPAQQ AIPSTODRAWRECORDS (PointRecord))
[DECLARE: EVAL@COMPILE

(RECORD PointRecord (xCoord . yCoord)
                    xCoord _ 0 yCoord _ 0)
]
)

(RPAQQ AIPSTODRAWUTILITYFNS (DisplaySurfaceForGroundP
                            FindDrawingContextsForDisplay
                            FindRoleValuesTransitively
                            FindTopLevelDisplaysOnViewSurface
                            GetFontCharacterDef
                            LDIFFERENCE ROUND
                            RotatePointsAroundPoint
                            RoundPoint
                            RoundPoints
                            ScrollWindow
                            TransformPoints
                            TransformSize))

(DEFINEQ

{ 27}

(DisplaySurfaceForGroundP
[LAMBDA (displaySurface ground)
  (* Edited by J.Gibbons on
  19-Jan-81.)
  (OR ground=displaySurface (AND ~(ground df `DISPLAYSURFACE)
                                  (for window in {ground;Window}
                                    thereis (CAR
                                              {window;Aperture;DisplaySurface})
                                              =displaySurface))

{ 28}

```

```
(FindDrawingContextsForDisplay
  [DLAMBDA ((display IndividualConcept (SATISFIES display df `DISPLAY)))
    (* Edited by J.Gibbons on
      28-Jan-81.)
```

(* Returns a list of the relevant drawing contexts for the Display and all of its subDisplays.)

```
(bind ground {display;&Ground}:1
  genericDisplay `DISPLAY
  drawingContexts windows for realization
  in {display;Realization} when realization df genericDisplay
  join (FindDrawingContextsForDisplay realization)
  finally (RETURN (INTERSECTION drawingContexts_
    <
      ! (if windows {ground;Window}
        then < T windows>
        elseif ground df `DISPLAYSURFACE
        then <ground>)
      ! $$VAL>
      drawingContexts))))))
```

{29}

```
(FindRoleValuesTransitively
  [DLAMBDA ((individualConcept IndividualConcept)
    (genericRole Role)
    (RETURNS (LST OF RoleValue)))
    (* Edited by J.Gibbons on
      27-Aug-80.)
```

(* Finds all of the individuals which are transitively related to the individualConcept by virtue of being a Role Filler of the genericRole. For example, if concept FOO has generic role BAR and individuaters A, B, and C such that B fills the BAR role of A and C fills the BAR role B, then both B and C are returned for A and BAR.)

```
(PROG (fillerLst)
  (RETURN < ! (fillerLst_ (KLFindRoleValues individualConcept
    genericRole))
    ! (for filler in fillerLst
      join (FindRoleValuesTransitively filler genericRole)
    )
  >)))
```

{ 30 }

```
(FindTopLevelDisplaysOnViewSurface
[DLAMBDA ((viewSurface IndividualConcept (SATISFIES (viewSurface df
                                                    ^VIEWSURFACE)))
  (RETURNS (LST OF IndividualConcept)))
  (* Edited by J.Gibbons on
    29-Jul-80.)
```

(* This returns a list of Displays each of which 1: has its
Ground role filled by viewSurface and 2: does not fill the
Realization role of some other Display having viewSurface as
Ground.)

```
[PROG (displays)
  (displays_(bind (groundRole _ (KLFindOneNamedGenericRole ^DISPLAY
                                                                'Ground))
    for groundIRole in (KLGetValueDescriptionInverses
                      viewSurface)
    collect (KLGetConceptOfRole groundIRole)
    when (KLIsRoleDescendantP groundIRole groundRole)))
  (RETURN (LDIFFERENCE displays
    (for display in displays
      join (FindRoleValuesTransitively
        display
        (KLFindOneNamedGenericRole
          ^DISPLAY
          'Realization]))
```

{ 31 }

```
(GetFontCharacterDef
[DLAMBDA ((fontNumber SMALLP (SATISFIES fontNumber ge 0
                                         and fontNumber le
                                         BMGMaxNumberOfFonts))
  (characterCode SMALLP (SATISFIES characterCode ge 0
                                   and characterCode le 127)))
  (* Edited by J.Gibbons on
    18-Sep-80.)
  (* Returns the BMGCharDefRecord
    for the specified characterCode
    in the specified font.)
'ELT (BMGDescribeFont fontNumber):LoadedFont\FontRecord:Font\CharArray
1+characterCode)])
```

{ 32 }

```
(LDIFFERENCE
[LAMBDA (LIST1 LIST2)
  (* Edited by J.Gibbons on
```

31-Jan-81.)

(* WE NEED THIS SINCE IT IS
UNDEFINED IN THE SMALL CKLONE)

(for X in LIST1 collect X unless X memb LIST2])

{33}

(ROUND
[DLAMBDA ((x NUMBERP))(* Edited by J.Gibbons on
7-Jan-81.)(if (FIXP x)
then x
else (FIX (if (MINUSP x)
then x-.5
else x+.5))))))

{34}

(RotatePointsAroundPoint
[DLAMBDA ((pointsLst (LST OF (LISTP OF NUMBERP)))
(rotationSpec (ONEOF NUMBERP LST)))
(CLISP: MIXED)(* Edited by J.Gibbons on
8-Jan-81.)(* Rotates a list of points counterclockwise from the positive
xAxis around a center point according to orientation.
Assumes a cartesian coordinate system.)(if rotationSpec and pointsLst
then (PROG ((theta (if (NUMBERP rotationSpec)
then rotationSpec
else rotationSpec:1))
[center (if (LISTP rotationSpec)
then rotationSpec:2
else (CONSTANT (create PointRecord)
cos sin xConstant yConstant)
(RETURN (if theta and ~(EQP theta 0)
then (cos (COS theta))
(sin (SIN theta))
(xConstant_ center:xCoord*(1.0-cos)
+center:yCoord*sin)
(yConstant_ center:yCoord*(1.0-cos)
-center:xCoord*sin)
(for point in pointsLst
collect (create PointRecord xCoord_
xConstant+point:xCoord*cos-point:yCoord*sin
yCoord_

```

                                yConstant+point:xCoord*sin+point:yCoord*cos))
                                else pointsLst)))
else pointsLst)))

```

{ 35 }

```

(RoundPoint
[LAMBDA (point)

```

```

(* Edited by J.Gibbons on
8-Jan-81.)

```

```

(* ROUNDS the coordinates of a point if either of them is a
FLOATP)

```

```

(if (FLOATP point:xCoord) or (FLOATP point:yCoord)
  then (create PointRecord
        xCoord (ROUND point:xCoord)
        yCoord (ROUND point:yCoord))
  else point))

```

{ 36 }

```

(RoundPoints
[DLAMBDA ((pointsLst (LST OF (LISTP OF NUMBERP))))

```

```

(* Edited by J.Gibbons on
9-Jan-81.)

```

```

(* Round all of the coordinates
of points in pointsLst to
integers.)

```

```

(for point in pointsLst collect (RoundPoint point)))

```

{ 37 }

```

(ScrollWindow
[DLAMBDA ((scrollWindow IndividualConcept (SATISFIES (scrollWindow df
`SCROLLWINDOW)))

```

```

(substrateLocation LISTP))

```

```

(CLISP: MIXED)

```

```

(* Edited by J.Gibbons on
1-Aug-80.)

```

```

(* Sets the Substrate Location
of the ScrollWindow and redraws
the Aperture.)

```

```

(KLChangeRoleValue (KLFindOneNamedInstanceRole scrollWindow
'SubstrateLocation)

```

```

<substrateLocation>)

```

```

(APPLY* (KLFindIData scrollWindow '(ToDrawAperture)):1 scrollWindow)])

```

{ 38 }


```

(TransformPoints
  [DLAMBDA ((pointsLst (LST OF (LISTP OF NUMBERP)))
            (rotationSpec (ONEOF LST NUMBERP))
            (magnification (ONEOF NIL NUMBERP))
            (translation (ONEOF NIL (LISTP OF NUMBERP))))
    (CLISP: MIXED)
    (* Edited by J.Gibbons on
    13-Jan-81.)
    (* Provides a uniform
    transformation mechanism for
    the ToDraw functions.)

    (if magnification
      else magnification_1)
    (if translation
      else translation (CONSTANT (create PointRecord)))
    (for point in (RotatePointsAroundPoint pointsLst rotationSpec)
      collect (create PointRecord xCoord
                                translation:xCoord+magnification*point:xCoord
                                yCoord
                                translation:yCoord+magnification*point:yCoord))))

```

{ 39 }

```

(TransformSize
  [DLAMBDA ((size NUMBERP)
            (magnification (ONEOF NIL NUMBERP)))
    (CLISP: MIXED)
    (* Edited by J.Gibbons on
    13-Jan-81.)
    (* Transforms the size
    parameter by magnification.)

    (if magnification
      then magnification*size
      else size))]
)
(DECLARE: DOEVAL@COMPILE DONTCOPY

(ADDTOVAR GLOBALVARS TDRegion)
)
(DECLARE: DONTVAL@LOAD EVAL@COMPILEWHEN (NOT (BOUNDP (QUOTE BMGCOMS)))
DONTCOPY
(FILELOAD <FONTWORK>BMGRECORDSANDVARS..0)
)

(ADDTOVAR CKLONEFILES AIPSTODRAW)
STOP

```

Bolt Beranek and Newman Inc.

Report No. 4752

15. SOURCE FILE: AIPSTS

InitTIMESYSTEM.....1
InitTimeSystemConcepts.....2
InitUT.....3

(FILECREATED "10-Jul-80 23:58:18" <NEWAIPS>AIPSTS..2 1129

changes to: AIPSTSCOMS

previous date: " 9-Jul-80 14:54:20" <NEWAIPS>AIPSTS..1)

(PRETTYCOMPRINT AIPSTSCOMS)

(RPAQQ AIPSTSCOMS ((FNS * AIPSTSINITFNS)
(ADDVARS (CKLONEFILES AIPSTS))))

(RPAQQ AIPSTSINITFNS (InitTIMESYSTEM InitTimeSystemConcepts InitUT))
(DEFINEQ

{1}

(InitTIMESYSTEM

[LAMBDA NIL

[concept TIMESYSTEM

(* A coordinate system for
describing events.)

(specializes COORDINATESYSTEM)

[roleset NIL

(mods Dimensionality@COORDINATESYSTEM)

(vr 1)]]])

{2}

(InitTimeSystemConcepts

[LAMBDA NIL

(* Initializes concepts having to do with describing time
intervals and the timings of events.)

(InitTIMESYSTEM])

{3}

(InitUT

[LAMBDA NIL

(* Edited by F.Zdybel on
9-Jul-80.)

[concept UT

(specializes TIMESYSTEM)

[roleset NIL

```
(mods Name@CoordinateSystem)
(vr "Universal Time")]]])
)
```

```
(ADDTOTVAR CKLONEFILES AIPSTS)
STOP
```


Report No. 4752

Bolt Beranek and Newman Inc.

16. SOURCE FILE: AIPSUTILITY

SendMessage....1
SetRoleValues..2

(FILECREATED "19-Dec-80 00:34:20" <NEWAIPS>AIPSUTILITY..9 2943

changes to: SendMessage

previous date: "16-Dec-80 23:54:51" <NEWAIPS>AIPSUTILITY..8)

(PRETTYCOMPRINT AIPSUTILITYCOMS)

```
(RPAQQ AIPSUTILITYCOMS [(FNS * AIPSUTILITYFNS)
  (VARS (SENDMESSAGENOBREAKFLG)
        (SENDMESSAGENOERRMESSFLG))
  (DECLARE: DONTEVAL@LOAD DOEVAL@COMPILE DONTCOPY
    COMPILERVERS (ADDVARS (NLAMA)
                          (NLAML)
                          (LAMA SendMessage]))
```

```
(RPAQQ AIPSUTILITYFNS (SendMessage SetRoleValues))
(DEFINEQ
```

{1}

```
(SendMessage
[LAMBDA argList
```

(* Edited by J.Gibbons on
19-Dec-80.)

(* Attached procedures on IndividualConcepts are invoked via
SendMessage (individualConcept procedureTag . args).
Procedures inherited by individualConcept according to the
iTag procedureTag are sequentially executed until one
succeeds, in which case SendMessage passes back the value
returned by the inherited procedure. If none succeed, an ERROR
is generated which can be controlled by the variables
SENDMESSAGENOBREAKFLG and SENDMESSAGENOERRMESSFLG.
An inherited procedure is APPLY'd to the same argument list as
SendMessage except that procedureTag is removed.)

```
(bind value (procedureArgList _ (for index to argList unless index=2
                                collect (ARG argList index)))
  for attachedProcedure in (KLFindIData (the Concept (ARG argList 1))
                                     (MKLIST (ARG argList 2)))
  thereis (NLSETQ value_ (APPLY attachedProcedure procedureArgList))
  finally (if $$VAL
    then (RETURN value)
    else (ERROR (ARG argList 1)
                (CONCAT "doesn't know how to process "

```



```
(ARG argList 2))
SENDMESSAGENOBREAKFLG SENDMESSAGENOERRMESSFLG])
```

```
{2}
```

```
(SetRoleValues
  [DLAMBDA ((iConcept IndividualConcept)
    (parentRole RoleSet)
    (roleValues (LST OF RoleValue)))
    (* Edited by J.Gibbons on
    16-Dec-80.)

  (* This function establishes a new set of role fillers for the
  parentRole of the iConcept. It changes role values of existing
  instance roles, deletes excess instance roles if any, or adds
  instance roles if necessary.)

  (bind (instanceRoles (KLFindInstanceRoles iConcept parentRole))
    for roleValue in roleValues do (if instanceRoles
      then (KLChangeRoleValue
        instanceRoles:1
        <roleValue>)

        instanceRoles instanceRoles::1
      else (KLSatisfyRole parentRole
        iConcept
        <roleValue>))

    finally (for instanceRole in instanceRoles
      do (KLRemoveRole instanceRole))))))

)

(RPAQ SENDMESSAGENOBREAKFLG NIL)

(RPAQ SENDMESSAGENOERRMESSFLG NIL)
(DECLARE: DONTEVAL@LOAD DOEVAL@COMPILE DONTCOPY COMPILERVERS

(ADDTOVAR NLAMA )

(ADDTOVAR NLAML )

(ADDTOVAR LAMA SendMessage)
)
STOP
```


17. SOURCE FILE: AIPSWINDOW

InitAPERTURE.....1
InitFASTWINDOW.....2
InitNONSCROLLWINDOW...3
InitSCROLLWINDOW.....4
InitSLOWWINDOW.....5
InitTTYWINDOW.....6
InitWINDOW.....7
InitWindowConcepts...8
OpenWindow.....9
TLAperture.....10
TLWindow.....11
TMAperture.....12
TMFastWindow.....13
TMNonScrollWindow....14
TMSlowWindow.....15
TMTTYWindow.....16
TSAperture.....17
TSNonScrollWindow....18
TSWindow.....19

(FILECREATED " 2-Feb-81 01:15:28" <NEWAIPS>AIPSWINDOW..122 18798

changes to: TMFastWindow

previous date: " 2-Feb-81 01:00:00" <NEWAIPS>AIPSWINDOW..121)

(PRETTYCOMPRINT AIPSWINDOWCOMS)

```
(RPAQQ AIPSWINDOWCOMS ((FNS * AIPSWINDOWINITFNS)
                        (FNS * AIPSWINDOWUTILITYFNS)
                        (FNS * AIPSWINDOWTOLOCATEFNS)
                        (FNS * AIPSWINDOWTOMAKEFNS)
                        (FNS * AIPSWINDOWTOSIZEFNS)
                        (GLOBALVARS AIPSUSERMODEL)
                        (ADDVARS (CKLONEFILES AIPSWINDOW))))
```

```
(RPAQQ AIPSWINDOWINITFNS (InitAPERTURE InitFASTWINDOW InitNONSCROLLWINDOW
                          InitSCROLLWINDOW InitSLOWWINDOW
                          InitTTYWINDOW InitWINDOW
                          InitWindowConcepts))
```

(DEFINEQ

{1}

(InitAPERTURE
[LAMBDA NIL

(* Edited by J.Gibbons on
30-Dec-80.)

```
[concept APERTURE
 (specializes DISPLAYCOMPOSITE)
 [roleset Border
  (diffs Component@DISPLAYCOMPOSITE)
  (modality Obligatory)
  (vr RECTANGLE)
  (derivation (TMRectangle))]]
 [roleset DisplaySurface
  (diffs Component@DISPLAYCOMPOSITE)
  (modality Obligatory)
  (vr DISPLAYSURFACE)]
 (itags (ToLocate TLApture)
        (ToMake TMApture)
        (ToSize TSApture))]]]
```

{2}

(InitFASTWINDOW

[LAMBDA NIL

(* Edited by J.Gibbons on
28-Jan-81.)

```
[concept FASTWINDOW
  (specializes SCROLLWINDOW)
  [roleset NIL
    (mods Substrate@SCROLLWINDOW)
    (vr INVISIBLESURFACE)
    (derivation (SeekInvisibleSubstrate
                  (Prerequisites Application)
                  (Arguments { (Make DISPLAY Application
                               ({ $$CONCEPT;Application}:1))
                               ;Ground}:1)))]]
```

(* Any Window whose substrate is an actual chunk of map memory
can scroll very quickly via BMGCopyRegion.
Hence the name FastWindow.)

```
(itags (Scroll ScrollFastWindow)
  (ToDraw TDFastWindow)
  (ToDrawInWindow TDIWFastWindow)
  (ToMake TMFastWindow)))]]
```

{3}

[InitNONSCROLLWINDOW
[LAMBDA NIL(* Edited by J.Gibbons on
28-Jan-81.)

```
[concept NONSCROLLWINDOW
  (specializes WINDOW)
```

(* The only distinguishing structural characteristic of
NonScrollWindows at this time is that their Substrates are
identical with the DisplaySurfaces of their Apertures.)

```
[roleset NIL
  (mods Aperture@WINDOW)
  (derivation (TMAperture (Prerequisites Substrate)
                          (Arguments NIL { $$CONCEPT;Substrate}:1))))
(itags (ToDraw TDNonScrollWindow)
  (ToDrawInWindow TDIWNonScrollWindow)
  (ToMake TMNonScrollWindow)
  (ToSize TSNonScrollWindow)))]]
```

{4}

```
(InitSCROLLWINDOW
[LAMBDA NIL
```

```
(* Edited by J.Gibbons on
23-Nov-80.)
```

```
[concept SCROLLWINDOW
(specializes WINDOW)
[roleset ScrollBar
(diff Realization@DISPLAY)
(number (0 2))
(vr CONTROLBAR)]
```

```
(* Designations on the Scroll Bar of a Window can be used to
modify the description filling the Window's Location Role.)
```

```
[roleset SubstrateLocation
(vr LISTP)
(defaultfiller '(0 0))]
```

```
(* The location of the Window's origin
(the origin of its display region) on the coordinate system of
the Substrate.)
```

```
(preferredprototype `FASTWINDOW))])
```

{5}

```
(InitSLOWWINDOW
[LAMBDA NIL
```

```
(* Edited by J.Gibbons on
20-Jan-81.)
```

```
[concept SLOWWINDOW
```

```
(* A Window which scales and scrolls by causing the stuff
being viewed through it to be re-Presented.)
```

```
(specializes SCROLLWINDOW)
[roleset ScaleBar
(diff Realization@DISPLAY)
(vr CONTROLBAR)]
```

```
(* Designations in the Scale Bar of a Window can be used to
modify the descriptions filling the Entry and Exit Roles of
the CoordSys of Window's Substrate (i.e., the scale of the
projection through the Window). Since there is no hardware
level stuff for doing scaling, it must be accomplished by
```

re-presentation, hence can only be accomplished by SlowWindows.)

```
(itags (Scroll ScrollSlowWindow)
      (ToDraw TDSlowWindow)
      (ToDrawInWindow TDIWSlowWindow)
      (ToMake TMSlowWindow)))]])
```

{6}

```
(InitTTYWINDOW
 [LAMBDA NIL
```

(* Edited by J.Gibbons on
27-Jan-81.)

```
[concept TTYWINDOW
```

(* Individuals of TTYWindow provide the Substrates for the BMGTTYRegion, ie. the destination of system TTY output.)

```
(specializes SCROLLWINDOW)
[roleset NIL
 (mods Substrate@SCROLLWINDOW)
 (derivation (Copy (Prerequisites Aperture)
                  (Arguments {$$CONCEPT;Aperture;&DisplaySurface}:1)
                ))]
(itags (Scroll ScrollTTYWindow)
      (ToDraw TDTTYWindow)
      (ToMake TMTTYWindow)))]])
```

{7}

```
(InitWINDOW
 [LAMBDA NIL
```

(* Edited by J.Gibbons on
27-Jan-81.)

```
[concept WINDOW
```

(* Window is a concept that serves many purposes. In the first place, a Window is a convenient way to organize and mobilize the use of the limited available display area. In the second place, it can serve as a context for the interpretation of input. Finally, a Window constitutes an important part of the environment for the realization of a display. This explains how it is possible to implement some scrolling in a way that eliminates undesirable edge effects: the "position" of the Window over the substrate can influence the re-layout and re-realization of the underlying

Presentations.)

(specializes DISPLAY)

(* The Context is the VisibleSurface on which the Window gets displayed. It must either be specified or defaulted.)

```
[roleset Context
  (mods Ground@DISPLAY)
  (vr DISPLAYSURFACE)
  (defaultfiller `CONRACSCREEN)]
```

(* The Substrate is the ViewSurface on which the Window opens. Currently there can be only one. Perhaps in the future we will allow more in which case the images will be superimposed. The Substrate must be specified.)

```
[roleset Substrate
  (modality Obligatory)
  (vr VIEWSURFACE)]
```

(* The Aperture is the VisibleSurface on which the portion of the Substrate covered by the Window is displayed.)

```
[roleset Aperture
  (diffs Realization@DISPLAY)
  (modality Obligatory)
  (vr APERTURE)
  (derivation (TMAperture (Prerequisites Context)
    (Arguments NIL { $$CONCEPT;Context;&Plane}:1
      T)))]
```

(* The Border is a Rectangle which also acts as the Boundary of the Window's DisplaySurface. It gets drawn to pictorially delimit the extent of the Window.)

```
[roleset Border
  (diffs Realization@DISPLAY)
  (vr RECTANGLE)
  (modality Obligatory)
  (derivation (Copy (Prerequisites DisplaySurface)
    (Arguments { $$CONCEPT;DisplaySurface;&Boundary}:1
```


)))

(* The DisplaySurface is all and only the display region on which the Window and all of its parts are drawn.)

```
[roleset DisplaySurface
  (diffs Realization@DISPLAY)
  (vr DISPLAYSURFACE)
  (modality Obligatory)
  (derivation (TMDisplaySurface (Prerequisites Context)
                                (Arguments NIL
                                   { $$CONCEPT;Context;&Plane}:1 0)))
```

]

(* The Label is an optional part of the Window nestled either above or below the Aperture inside the Border. It is manifest by a special sort of Window which itself has no Label but is drawn as Text on an inverted background.)

```
[roleset Label
  (diffs Realization@DISPLAY)
  (vr WINDOWLABEL)
  (derivation (MakeWindowLabel (Arguments $$CONCEPT))
              (Consequents InnerServant))]
```

(* The Location of the Window is the position of its lower left corner on its Context.)

```
[roleset NIL
  (mods Location@DISPLAY)
  [derivation (MakeWindowLocation (Prerequisites Context Height Width)
                                  (Arguments (CAR { $$CONCEPT;Context})
                                              (CAR { $$CONCEPT;Height})
                                              (CAR { $$CONCEPT;Width})))]
```

(* The Width of the Window can be derived from any of several other roles depending on what is already filled. We will give as arguments all of the possibilities and let the derivation function determine what to do.)

```
[roleset NIL
  (mods Width@DISPLAY)
  [derivation (MakeWindowWidth (Arguments (CAR
```

```

    {$$CONCEPT;Aperture;Width}}
    (CAR
    {$$CONCEPT;Border;Width}}
    (CAR
    {$$CONCEPT;Substrate;Width}})]

```

(* The Height of the Window can be derived from any of several other roles depending on what is already filled. We will give as arguments all of the possibilities and let the derivation function determine what to do. In addition, the Label can affect the Height of the Window if it is present.)

```

[roleset NIL
  (mods Height@DISPLAY)
  [derivation (MakeWindowHeight (Arguments (CAR
                                           {$$CONCEPT;Aperture;Height}}
                                           (CAR
                                           {$$CONCEPT;Border;Height}}
                                           (CAR
                                           {$$CONCEPT;Substrate;Height}}
                                           (CAR
                                           {$$CONCEPT;Label;Height}})]

[roleset NIL
  (mods Envelope@DISPLAY)]
[PROG (servant)
  (servant_ [roleset Servant
              (number (0 NIL))
              (vr WINDOW)])]

```

(* A Window may have other windows closely associated with it. The "Servant" relationship indicates other windows whose locations must be attended to when a window is moved.)

```

[roleset OuterServant
  (diffs (atomval servant))]
[roleset InnerServant
  (diffs (atomval servant))
  (derivation (CopyWindowInnerServantFromLabel
               (Arguments {$$CONCEPT;Label}:1)))]

```

(* The role Servant is differentiated into InnerServant and OuterServant because the rules for relocating these two cases are likely to differ. Note that it is necessary to use a circumlocution in order to set up the differentiation properly (because of name inheritance.)

```

[roleset Master
  (vr WINDOW)]
(preferredprototype `NONSCROLLWINDOW)
(itags (ToDraw TDWindow)
       (ToLocate TLWindow)
       (ToSize TSWindow)))]])

```

{8}

```

(InitWindowConcepts
  [LAMBDA NIL

```

```

(* Edited by J.Gibbons on
27-Jan-81.)

```

```

  (InitAPERTURE)
  (InitWINDOW)
  (InitNONSCROLLWINDOW)
  (InitSCROLLWINDOW)
  (InitTTYWINDOW)
  (InitSLOWWINDOW)
  (InitFASTWINDOW)]
)

```

```

(RPAQQ AIPSWINDOWUTILITYFNS (OpenWindow))
(DEFINEQ

```

{9}

```

(OpenWindow
  [DLAMBDA ((windowType Concept (SATISFIES windowType df `WINDOW))
            (substrate IndividualConcept (SATISFIES substrate df
            `VIEWSURFACE)))
    (* Edited by J.Gibbons on
    29-Jan-81.)

```

```

  (PROG (window)
    (window_ (SendMessage windowType 'ToMake substrate):1)
    (SendMessage window 'ToSize)
    (SendMessage window 'ToLocate)
    (KLAddInstanceRole substrate Window@VIEWSURFACE <window>)
    (RETURN window)))
)

```

```

(RPAQQ AIPSWINDOWTOLOCATEFNS (TLAperture TLWindow))
(DEFINEQ

```

{10}

```

(TLAperture
  [DLAMBDA ((aperture IndividualConcept (SATISFIES aperture df `APERTURE))

```

```
(location (LST OF FIXP)))
```

```
(* Edited by J.Gibbons on
29-Jan-81.)
```

```
(* The location of the Aperture itself must be
(0 0) since it is a DisplayComposite. Otherwise, the location
would translate the location of its parts which would be
disastrous for the DisplaySurface.)
```

```
(if location or ~location_{aperture;Border;Location}:1
  then (if location
        then (SetRoleValues aperture Location@APERTURE '((0 0)))
        else location_{aperture;&Border;&Location}:1)
    (SendMessage {aperture;&Border}:1 'ToLocate location)
    (SendMessage (CAR {aperture;&DisplaySurface})
      'ToLocate <location:1+1 location:2+1>))
location])
```

```
{11}
```

```
(TLWindow
```

```
[DLAMBDA ((window IndividualConcept (SATISFIES window df `WINDOW))
  (location (LST OF FIXP)))
```

```
(* Edited by J.Gibbons on
29-Jan-81.)
```

```
(if location or ~location_{window;Location}:1
  then (if location
        else location_ <(bind xCoord
                          do (printout T T "Window xCoord? ")
                          (xCoord (RATOM))
                          repeatuntil (FIXP xCoord)
                          finally (RETURN xCoord_(xCoord+8)/16*16))
        (bind yCoord
              do (printout T T "Window yCoord? ")
              (yCoord (RATOM))
              repeatuntil (FIXP yCoord)
              finally (RETURN yCoord))
        >)
    (SetRoleValues window Location@WINDOW <location>)
    (SendMessage {window;&DisplaySurface}:1 'ToLocate location)
    (SendMessage (CAR {window;&Aperture})
      'ToLocate <location:1+15 location:2>))
location])
)
```

```
(RPAQQ AIPSWINDOWTOMAKEFNS (TMAperture TMFastWindow TMNonScrollWindow
  TMSlowWindow TMTTYWindow))
```

```
(DEFINEQ
```

{12}

```

(TMAperture
  [LAMBDA (genericAperture displaySurface planeFlg)
    (* Edited by J.Gibbons on
      27-Jan-81.)
    <(Make APERTURE DisplaySurface (if planeFlg
      then (TMDisplaySurface NIL
        displaySurface 0)
      :1
      else displaySurface)
    Border 'Fill)
  >])

```

{13}

```

(TMFastWindow
  [DLAMBDA ((genericFastWindow GenericConcept (SATISFIES genericFastWindow
    df `FASTWINDOW))
    (substrate IndividualConcept (SATISFIES substrate df
      `DISPLAYSURFACE)))
    (* Edited by J.Gibbons on
      2-Feb-81.)
    <(Make FASTWINDOW Substrate substrate Context 'Fill DisplaySurface
      'Fill Border 'Fill Aperture 'Fill)
    >])

```

{14}

```

(TMNonScrollWindow
  [LAMBDA (genericNonScrollWindow substrate)
    (* Edited by J.Gibbons on
      27-Jan-81.)
    <(Make NONSCROLLWINDOW Substrate substrate Context 'Fill DisplaySurface
      'Fill Border 'Fill Aperture 'Fill)
    >])

```

{15}

```

(TMSlowWindow
  [LAMBDA (genericSlowWindow substrate)
    (* Edited by J.Gibbons on
      18-Dec-80.)
    <(Make SLOWWINDOW Substrate substrate Context 'Fill DisplaySurface
      'Fill Border 'Fill Aperture 'Fill)
    >])

```

{16}

```

(TMTTYWindow
[DLAMBDA ((genericTTYWindow GenericConcept (SATISFIES genericTTYWindow df
      ^TTYWINDOW))
      (context IndividualConcept (SATISFIES context df
      ^DISPLAYSURFACE)))
      (* Edited by J.Gibbons on
      2-Jan-81.)

(* Derives a NonScrollWindow for displaying TTY output.
Yes, this is a misnomer in that scrolling will occur but will
be entirely under control of BMG. This may be just one
indication that the hierarchy of windows needs rethinking.)

<(Make TTYWINDOW Context context DisplaySurface 'Fill Border 'Fill
Aperture 'Fill Substrate 'Fill)
>])
)

(RPAQQ AIPSWINDOWTOSIZEFNS (TSAperture TSNonScrollWindow TSWindow))
(DEFINEQ

```

{17}

```

(TSAperture
[DLAMBDA ((aperture IndividualConcept (SATISFIES aperture df ^APERTURE))
      (height (ONEOF NIL FIXP))
      (width (ONEOF NIL FIXP)))
      (* Edited by J.Gibbons on
      28-Jan-81.)

(if height or width or ~(height_{aperture;Height}:1
      and width_{aperture;Width}:1)
    then (if height
      then (SetRoleValues aperture Height@APERTURE <height>)
      else height_{aperture;&Height}:1)
    (if width
      then (SetRoleValues aperture Width@APERTURE <width>)
      else width_{aperture;&Width}:1)
    (SendMessage {aperture;&Border}:1 'ToSize height width)
    (SendMessage {aperture;&DisplaySurface}:1 'ToSize height-2
      width-2))
    <height width>])

```

{18}

```

(TSNonScrollWindow
[DLAMBDA ((nonScrollWindow IndividualConcept (SATISFIES nonScrollWindow

```

```
df
`NONSCROLLWINDOW))
```

```
(height (ONEOF NIL FIXP))
(width (ONEOF NIL FIXP))
```

```
(* Edited by J.Gibbons on
29-Jan-81.)
```

(* Sets the Height and Width if the argument is supplied. Otherwise, they are derived if not already set. Setting of the values causes ToSize messages to be sent to the nonScrollWindow's parts. In any case, the values are returned.)

```
(PROG ((substrate {nonScrollWindow;&Substrate}:1))
  (if height or width or ~(height_{nonScrollWindow;Height}:1
    and width_{nonScrollWindow;Width}:1)
    then (if height
      else height_2+{substrate;&Height}:1)
      (if width
        else width_32+{substrate;&Width}:1)
      (SetRoleValues nonScrollWindow Height@NONSCROLLWINDOW
        <height>)
      (SetRoleValues nonScrollWindow Width@NONSCROLLWINDOW
        <width>)
      (SendMessage {nonScrollWindow;&Aperture}:1 'ToSize
        height width-30)
      (SendMessage {nonScrollWindow;&DisplaySurface}:1 'ToSize
        height width))
    (RETURN <height width>))))
```

{19}

```
(TSWindow
  [DLAMBDA ((window IndividualConcept (SATISFIES window df `WINDOW))
    (height (ONEOF NIL FIXP))
    (width (ONEOF NIL FIXP)))
```

```
(* Edited by J.Gibbons on
29-Jan-81.)
```

(* Sets the Height and Width if the argument is supplied. Otherwise, they are derived if not already set. Setting of the values causes ToSize messages to be sent to the window's parts. In any case, the values are returned.)

```
(if height or width or ~(height_{window;Height}:1
  and width_{window;Width}:1)
  then (if height
    else (do (printout T T "Window height? ")
```

```
        (height_(RATOM)) repeatuntil (FIXP height)))
    (if width
      else (do (printout T T "Window width? ")
        (width_(RATOM)) repeatuntil (FIXP width)
        finally width_(width+8)/16*16))
    (SetRoleValues window Height@WINDOW <height>)
    (SetRoleValues window Width@WINDOW <width>)
    (SendMessage {window;&Aperture}:1 'ToSize height width-30)
    (SendMessage {window;&DisplaySurface}:1 'ToSize height width))
    <height width>])
)
(DECLARE: DOEVAL@COMPILE DONTCOPY
(ADDTOVAR GLOBALVARS AIPSUSERMODEL)
)

(ADDTOVAR CKLONEFILES AIPSWINDOW)
STOP
```